AGI-ORIENTED HYPERKNOWLEDGE BASE AS AN AGI OS

M-TAOS v1.0

نظام مساعف الفائق لتشغيل المهام الموجّه نحو الذكاء العام

Moussaif Task AGI-Oriented HyperOperating System

هندسة تشغيلية لتحويل مهام ChatGPT المجدولة والمحادثات الطويلة إلى أنظمة معرفية-وكيلية ذات ذاكرة، تفريع، تحقق، أدوات، حلقات تصحيح، أثر عالمي، وتراكم بين التشغيلات.

الإصدار
v1.0 — August 2026

المؤلف المعرفي
Zouhair Moussaif / زهير مساعف

مبدأ التشغيل
لا معرفة بلا تشغيل؛ لا تشغيل بلا تحقق؛ لا تحقق بلا ذاكرة وأثر

الوضع الإبستيمي
تركيب نظري-هندسي مؤسس على أدبيات الوكلاء والاستدلال والذاكرة والRAG

الملخص التنفيذي لنظام التشغيل
1. إعلان السلامة الإبستيمية وسجل المادة المتاحة
مراتب الادعاء المعتمدة
2. التصنيف الأنطولوجي للمجال
3. دستور المجال
3.1 الغرض
3.2 النطاق
3.3 تعريف النجاح والفشل
4. السؤال المركزي والتوتر والمتغير الخفي
5. النواة التوليدية Generative Kernel
6. المعجم التشغيلي الأساسي
7. الأساس البحثي: ماذا تضيف الأدبيات إلى Task OS؟
7.1 محكمة الادعاء: الحوسبة الأكثر ليست دائماً أفضل
8. النظريات والقوانين الجديدة في M-TAOS
8.1 قانون المنفعة الحدية للحوسبة — Marginal Compute Value Law
8.2 نظرية الكثافة الإنجازية الحوسبية — Computational Action Density
8.3 نظرية التوجيه الوعائي للحوسبة — Vascular Compute Routing
8.4 قانون حفظ الفرق الحاكم — Governing Difference Conservation
8.5 نظرية فرّع-اهدم-ادمج — Branch–Falsify–Merge Theory
8.6 قانون أسبقية الاختبار التنفيذي — Execution Proof Priority
8.7 نظرية ذاكرة إعادة البناء — Reconstruction Memory Theory
8.8 قانون انتشار المهارة عبر المهام — Cross-Task Skill Diffusion
8.9 نظرية تضاعف الأثر بالكتابة الفورية — Immediate Writeback Compounding
8.10 قانون تباعد الوكلاء النافع — Useful Agent Diversity
8.11 نظرية حدود التوازي — Parallelism Boundary Theory
8.12 قانون إغلاق العالم-الذاكرة — World-Memory Closure
8.13 نظرية السطح المتعدد للمخرج — Multi-Surface Output Theory
8.14 قانون الانفتاح الهجومي المنضبط — Disciplined Opening Law
8.15 نظرية الـHyperFrontier — Persistent Frontier Theory
8.16 قانون مقاومة الضجيج — Noise Subtraction Law
8.17 نظرية الميزانية متعددة العملات — Multi-Currency Compute Budget
8.18 قانون التحقق غير المتناظر — Asymmetric Verification
9. معادلات ومقاييس توجيه الحوسبة
9.1 MCD للمهمة
9.2 دالة التوجيه الوعائي
9.3 كثافة الأثر
9.4 معامل إعادة البناء
9.5 درجة MEGA/PETA
10. خريطة المفاهيم
11. خريطة المشكلات
12. خريطة الآليات
13. HyperSkills — قدرات قابلة للتدريب
13.1 HyperSkill: Kernel Extraction
13.2 HyperSkill: Goal-to-World-Delta Compilation
13.3 HyperSkill: Dependency Graphing
13.4 HyperSkill: Adaptive Compute Routing
13.5 HyperSkill: Evidence Packet Engineering

HyperSkill: Countermodel Generation 13.6
HyperSkill: Branch Diversity Control 13.7
HyperSkill: Falsification Routing 13.8
HyperSkill: Tool Capability Mapping 13.9
HyperSkill: Parallel Tool Orchestration 13.10
HyperSkill: Episodic Memory Distillation 13.11
HyperSkill: Semantic Memory Synthesis 13.12
HyperSkill: Procedural Skill Extraction 13.13
HyperSkill: Conflict-Preserving Merge 13.14
HyperSkill: Compute Saturation Detection 13.15
HyperSkill: Execution-Proof Design 13.16
HyperSkill: High-Agency Verification 13.17
HyperSkill: Canonical Ledger Engineering 13.18
HyperSkill: HyperFrontier Management 13.19
HyperSkill: Cross-Task Knowledge Diffusion 13.20
HyperSkill: False-Novelty Detection 13.21
HyperSkill: Research Attack Synthesis 13.22
HyperSkill: Opportunity-to-Outreach Compiler 13.23
HyperSkill: Artifact Density Engineering 13.24
HyperSkill: Source Freshness Routing 13.25
HyperSkill: Uncertainty Budgeting 13.26
HyperSkill: Self-Correction with External Signal 13.27
HyperSkill: Multi-Agent Role Orthogonality 13.28
HyperSkill: Run-Delta Compression 13.29
HyperSkill: Versioned Kernel Evolution 13.30
HyperSkill: Permission-Aware Agency 13.31
HyperSkill: Failure-Memory Encoding 13.32
HyperSkill: Benchmark Construction 13.33
HyperSkill: MCD Difference Scoring 13.34
HyperSkill: QMCD Model Superposition 13.35
HyperSkill: World-Effect Accounting 13.36
HyperSkill: Output Surface Compilation 13.37
HyperSkill: Mega-Grade Workstream Expansion 13.38
HyperSkill: PETA-Grade Runtime Arbitration 13.39
HyperSkill: Destruction-Court Operation 13.40

14. HyperLoops — حلقات التشغيل ذات الذاكرة

Kernel Extraction Loop 14.1
Evidence Ranking Loop 14.2
Concept Disambiguation Loop 14.3
Problem Decomposition Loop 14.4
Hypothesis Generation Loop 14.5
Destruction and Repair Loop 14.6
Tool Construction Loop 14.7
Skill Training Loop 14.8
Retrieval Optimization Loop 14.9
Agentic Execution Loop 14.10
Update and Versioning Loop 14.11
Research Program Loop 14.12
Compute Routing Loop 14.13
Parallel Branch Loop 14.14

14.15 Branch-Falsify-Merge Loop
14.16 World-Effect Loop
14.17 Memory Consolidation Loop
14.18 Failure Reconstruction Loop
14.19 Cross-Task Diffusion Loop
14.20 Opportunity Conversion Loop
14.21 Artifact Production Loop
14.22 Hallucination Suppression Loop
14.23 Saturation Loop
14.24 HyperFrontier Loop

15. طبقة الوكلاء Agentic Layer
16. طبقة الأدوات Tool Layer

16.1 Concept Kernelizer
16.2 Evidence Ranker
16.3 Difference Extractor
16.4 MCD Evaluator
16.5 QMCD Superposition Map
16.6 Task DAG Compiler
16.7 Vascular Compute Router
16.8 Parallel Dispatch Planner
16.9 Branch Diversity Meter
16.10 Destruction Court Matrix
16.11 HyperSkill Distiller
16.12 HyperLoop Orchestrator
16.13 RAG Chunking Compiler
16.14 GraphRAG Relation Compiler
16.15 Memory Write Filter
16.16 Failure Signature Detector
16.17 Execution Proof Harness
16.18 Outreach Dedupe Gate
16.19 Run Delta Compiler
16.20 HyperFrontier Ranker
16.21 AGI-Readiness Index
16.22 Prompt/Kernel Regression Tester
16.23 Cross-Task Diffusion Compiler
16.24 Compute Ledger

17. هندسة الذاكرة وRAG/GraphRAG

17.1 Chunk types
17.2 Metadata الدنيا
17.3 Graph nodes/edges
17.4 الاسترجاع المحلي والعالمي

18. مترجم التنفيذ Runtime Frontier Compiler
19. مستويات MEGA/PETA للتشغيل
20. محاكم الهدم

20.1 محكمة الهلوسة
20.2 محكمة الاختزال
20.3 محكمة الأصالة الزائفة
20.4 محكمة ضعف التشغيل
20.5 محكمة فشل الوكيل

20.6 محكمة تشبع الحوسبة
20.7 محكمة التكرار
20.8 محكمة الأثر
20.9 محكمة التوازي
20.10 محكمة الذاكرة

21. طبقة التقييم وAGI-Readiness
22. بروتوكول تشغيل مهمة واحدة

المرحلة 0 — State Load
المرحلة 1 — World Delta
المرحلة 2 — Graph Compile
المرحلة 3 — Compute Allocation
المرحلة 4 — Discovery/Execution
المرحلة 5 — Verification
المرحلة 6 — Branch-Falsify-Merge
المرحلة 7 — World Effect
المرحلة 8 — Commit
المرحلة 9 — Saturation Court

23. بروتوكول أسطول المهام Task Fleet OS
24. هندسة المهام الكبيرة: من Mega Prompt إلى Mega Execution
25. Deployment Modes
26. نظام النسخ والتحديث
27. برنامج بحثي لتطوير علم Task OS
28. اثنتا عشرة فرضية بحثية
29. اثنا عشر سؤالاً مستقبلياً عميقاً
30. اثنا عشر مفهوماً مفتاحياً يجب ضبطه
31. اثنا عشر خطأً شائعاً
32. اثنا عشر تمييزاً يغير قراءة المجال
33. النموذج الجديد: M-TAOS / M-PETA Architecture

33.1 لماذا نحتاجه؟
33.2 الطبقات
33.3 ما الذي يفسره أفضل؟
33.4 ما الذي لا يفسره؟
33.5 التنبؤات
33.6 اختبار النموذج

34. نسخة النموذج في صفحة واحدة
35. محكمة الهدم الكبرى: أقوى الاعتراضات

الاعتراض 1: إعادة تسمية معمارية الوكلاء
الاعتراض 2: المقاييس غير معايرة
الاعتراض 3: التوازي قد يزيد الأخطاء
الاعتراض 4: الذاكرة قد ترسخ الهلوسة
الاعتراض 5: self-critique غير موثوق
الاعتراض 6: multi-agent theater
الاعتراض 7: world-effect يرفع المخاطر
الاعتراض 8: تكلفة التنسيق
الاعتراض 9: ChatGPT tasks لها حدود منصة
الاعتراض 10: لا AGI هنا

35.1 عشر نقاط ضعف داخلية
35.2 النسخة المعدلة بعد الهدم

36. بروتوكول إدماج M-TAOS في مهام ChatGPT
36.1 Final AGI OS Kernel
M-TAOS v1.0 — COMPACT TASK KERNEL
37. سجل البناء المعرفي
38. التقييم الذاتي للجودة قبل التسليم
39. قائمة المصادر الأساسية
39.1 ملاحظة استعمال المصادر
40. الخلاصة التشغيلية
41. خمسون HyperRules كنظام دستوري تشغيلي
41.1 Goal Delta First
41.2 No Re-Discovery
41.3 State Before Search
41.4 Graph Before Scale
41.5 Parallelize Independence
41.6 Critical Path Priority
41.7 Uncertainty Must Be Reducible
41.8 Verification Scales With Consequence
41.9 External Signal Preference
41.10 Branch Diversity Is Functional
41.11 Merge by Evidence
41.12 Preserve Productive Disagreement
41.13 Action Needs Readback
41.14 Canonical Store Wins
41.15 Stable IDs Before Volume
41.16 Suppression Is Memory
41.17 Freshness Is Claim-Specific
41.18 Evidence Has Scope
41.19 Novelty Requires Operational Delta
41.20 Compression Must Regenerate
41.21 Memory Needs Forgetting
41.22 Failures Become Guardrails
41.23 Successes Become Skills
41.24 Cross-Task Transfer Needs Scope
41.25 Budget Is Multi-Currency
41.26 Stop on Marginal Collapse
41.27 Minimum Pass Floor
41.28 High Volume Requires Higher Courts
41.29 No Quantity Fabrication
41.30 Output Must Be Typed
41.31 Chat Is a View, Not the Database
41.32 Tool Choice Is a Reasoning Step
41.33 Interface Design Matters
41.34 Replan on State Change
41.35 Separate Discovery From Acceptance
41.36 Separate Activity From Outcome
41.37 Record Negative Evidence
41.38 Use Edge Cases as Probes
41.39 Use Inversion to Open Domains
41.40 Use Cross-Domain Fusion Carefully

Court Must Repair 41.41
Version Every Kernel Change 41.42
Regression Before Fleet Diffusion 41.43
Protect Specific Constraints 41.44
Frontier Must Be Ranked 41.45
Frontier Must Expire 41.46
Every Run Should Raise Starting Intelligence 41.47
AGI-Oriented Is Not AGI 41.48
Compute Claims Need Observable Work 41.49
The Highest Output Is Reusable Capability 41.50

42. أنماط تشغيل تطبيقية لأسطول المهام
Pattern: Job / Recruiter Sprint 42.1
Pattern: B2B Revenue Engine 42.2
Pattern: Research Collaboration Engine 42.3
Pattern: Reply / Inbox Monitor 42.4
Pattern: Carousel / Visual Production 42.5
Pattern: Creative Music Momentum 42.6
Pattern: Deep Research Book 42.7
Pattern: Software Engineering Agent 42.8
Pattern: Data Analysis Task 42.9
Pattern: Market / News Intelligence 42.10
Pattern: Scientific Literature Radar 42.11
Pattern: CRM / Archive Consolidation 42.12
Pattern: Talent Headhunter Outreach 42.13
Pattern: Long-Form Content Factory 42.14
Pattern: Knowledge Base Evolution 42.15
Pattern: General Opportunity Capture 42.16

43. Benchmark Suite لنظام Task OS
44. ثلاثون Failure Signatures وذاكرة الوقاية
45. مصفوفة Compute × Evidence × Agency
46. Knowledge Cores كنواة قابلة للتوسع
Core 1 — Goal Semantics
Core 2 — State Semantics
Core 3 — Evidence Semantics
Core 4 — Task Graph Semantics
Core 5 — Compute Economics
Core 6 — Memory Architecture
Core 7 — Tool Semantics
Core 8 — Agency Governance
Core 9 — Verification
Core 10 — Skill Learning
Core 11 — Fleet Governance
Core 12 — Versioning
Core 13 — Opportunity Systems
Core 14 — Artifact Systems
Core 15 — Research Opening
Core 16 — Epistemic Safety
Core 17 — Destruction
Core 18 — Evaluation
Core 19 — Cross-Task Intelligence

47. بروتوكول PETA لحدود التشغيل الواقعية
48. دستور كتابة المخرجات الضخمة
49. ملخص تنفيذي من صفحتين — النسخة المكثفة
الصفحة التنفيذية الأولى: لماذا Task OS؟
الصفحة التنفيذية الثانية: كيف يصبح الأسطول أذكى؟
Final AGI OS Kernel — النسخة القابلة للحقن
M-TAOS v1.0 — COMPACT TASK KERNEL

الملخص التنفيذي لنظام التشغيل

هذا الكتاب لا يقترح «برومبتا أطول» للمهام المجدولة، بل يقترح طبقة تشغيل معرفية تجعل المهمة نفسها وحدة حوسبة معرفية قابلة للتوسع والتصحيح والتراكم. المشكلة المركزية التي ينطلق منها النظام هي أن كثيرا من مهام الوكلاء الحديثة تُعامل كاستدعاء واحد لنموذج لغوي: سؤال ثم جواب، أو بحث ثم ملخص. هذه البنية تهدر ثلاثة مصادر للقوة: إمكانية توزيع الحوسبة وقت الاختبار على الفروع ذات القيمة الأعلى، إمكانية تحويل تاريخ المحادثة إلى ذاكرة تشغيلية لا إلى أرشيف خام، وإمكانية إغلاق الحلقة بين المعرفة والفعل والقياس والتحديث.

يقترح M-TAOS بنية من سبع طبقات مترابطة: نواة غائية تحدد ما الذي يجب أن يتغير في العالم؛ مترجم مهام يحول الطلب إلى رسم اعتماد (Task DAG/Graph)؛ موجه وعائي للحوسبة يوزع الميزانية على الفروع بحسب القيمة وعدم اليقين والمركزية والمخاطر؛ ذاكرة متعددة الطبقات تستخرج من المحادثات والملفات وسجل التشغيل ما يصلح لإعادة الاستخدام؛ طبقة أدوات وفعل تنفذ عمليات حقيقية عندما تكون مأذونة وآمنة؛ محاكم تحقق وتكذيب تمنع التضخم الخطابي والهلوسة والتكرار؛ ثم ذاكرة إعادة بناء تجعل نتيجة كل تشغيل مادة أولية لتشغيل أقوى لاحق.

ويعيد النظام تعريف عبارتي «MEGA» و«PETA» تعريفاً تشغيلياً حتى لا تتحولا إلى مبالغة كمية. MEGA تعني **Multi-pass Evidence-Grounded Agency**: وكالة متعددة المرور مؤسسة على الدليل. وPETA تعني **Parallel Epistemic Tool-Action**: تفريع متوازٍ معرفي-أداتي يهدف إلى إنتاج أكبر كمية مفيدة من النتائج المتحققة ضمن حدود الزمن والأدوات والتكلفة. لا يعد النظام بيتابايتات من النص، بل بمضاعفة كثافة القيمة لكل تشغيل: فروع مستقلة أكثر، تحقق أقوى، نتائج أكثر قابلية للفعل، ذاكرة أفضل، وتكرار أقل.

النظرية المركزية الجديدة هي **نظرية الكثافة الإنجازية الحوسبية** (Computational Action Density Theory): قيمة تشغيل المهمة لا تتناسب مع عدد التوكنات، بل مع مقدار التغير الموثق الذي ينتجه كل قدر من الحوسبة. لذلك يرفض M-TAOS مبدأ «فكر أكثر دائما» ويستبدله بمبدأ «زد الحوسبة ما دامت المنفعة الحدية موجبة ويمكن التحقق منها». هذا يتسق مع أدبيات test-time compute التي تبين أن تخصيص الحوسبة تكيفياً يمكن أن يفوق الاستخدام الساذج لعينات أكثر، ومع أدبيات الوكلاء التي تبين أن التخطيط، الذاكرة، الأدوات، واجهات التنفيذ، والتقييم التنفيذي تغير الأداء جذريا [S06][S15][S18][S23].

> **القانون الأعلى**
> المعرفة لا تكتمل حتى تتحول إلى قرار قابل للتنفيذ؛ والقرار لا يكتمل حتى يمر بمحكمة دليل؛ والمحكمة لا تكتمل حتى تحدث الذاكرة؛ والذاكرة لا تكتمل حتى ترفع قدرة التشغيل التالي.

1. إعلان السلامة الإبستيمية وسجل المادة المتاحة

بُني هذا الإصدار من ثلاثة أنواع من المادة. النوع الأول هو المادة التي قدمها المستخدم في هذا الطلب: دستور HyperKnowledge OS، مفاهيم MCD/QMCD/MCAI/HAKIM-MCD، بروتوكول «الباحث المحارب»، اشتراطات الـHyperSkills والـHyperLoops والمحاكم والـRAG/GraphRAG، وهدف دفع مهام ChatGPT إلى أقصى إنتاج مفيد في كل تشغيل. النوع الثاني هو المعرفة العامة المستقرة حول هندسة الوكلاء والنماذج اللغوية. النوع الثالث هو بحث ويب مباشر في بيانات وملخصات أوراق مفتوحة أو صفحات رسمية/بحثية حول Self-Refine، MemGPT، CoALA، LLMCompiler، AgentBench، GAIA، SWE-bench، OSWorld، GraphRAG، HippoRAG، test-time scaling وغيرها.

لم تُقرأ كل ورقة في القائمة المرجعية قراءة كاملة صفحة بصفحة في هذا التشغيل؛ جرى الاعتماد على الملخصات، بيانات النشر، صفحات الأوراق، وبعض المقاطع/النتائج المتاحة عبر البحث. لذلك أي نسبة أو نتيجة تجريبية مأخوذة من ملخص ورقة تُعامل كـ«معلومة منسوبة إلى الورقة» لا كإعادة تحليل مستقل للتجربة. أما MCD/QMCD ومشتقات M-TAOS الجديدة فهي **تركيب نظري** أو **فرضيات هندسية** وليست حقائق علمية مثبتة. لا يدعي هذا الكتاب أن M-TAOS يحقق AGI، بل أنه يبني بنية تشغيل موجهة نحو قدرات مرتبطة بالوكلاء العامين: التخطيط، الأدوات، الذاكرة، التحقق، التعلم من التاريخ، ونقل المهارة بين المهام.

مراتب الادعاء المعتمدة

الرتبة	المعنى	الاستخدام داخل الكتاب
Established / ثابت	مدعوم بأدبيات راسخة أو تعريف تقني مستقر	مفاهيم RAG، الرسوم البيانية، أدوات التقييم التنفيذي
Strongly supported / مدعوم بقوة	نتائج متكررة أو ورقة قوية مع أدلة واضحة	فائدة الأدوات، الذاكرة الخارجية، التخطيط في مهام كثيرة
Plausible / راجح	له أساس معقول لكنه حساس للمهمة/النموذج	فائدة التفريع أو الانعكاس في بعض البيئات
Contested / خلافي	الأدلة متعارضة أو تعتمد بقوة على البروتوكول	التصحيح الذاتي الداخلي، مناظرة الوكلاء
Theoretical synthesis / تركيب نظري	بناء جديد يجمع أدبيات ومفاهيم	قوانين M-TAOS ومقاييسه
Hypothesis / فرضية	اقتراح قابل للاختبار	فرضيات الكثافة الإنجازية، الانتشار عبر المهام
Speculative / تأملي	اتجاه غير مثبت	إسقاطات على AGI طويل الأفق
Needs verification / يحتاج تحقق	معلومة لم تتوفر أدلتها الكافية في هذا التشغيل	أي ادعاء دقيق غير مسند مباشرة

2. التصنيف الأنطولوجي للمجال

مجال «Task OS» ليس كياناً واحداً؛ هو في الوقت نفسه **عملية ونظام وشبكة ووكيل ومعمارية تقنية وبنية إبستيمية ومعمارية موجهة نحو AGI**. هذا التعدد ليس تجميلاً تصنيفياً. لو عولمت المهمة ككائن ثابت فسوف تختفي ديناميكياتها عبر الزمن؛ ولو عولمت كبرومبت فقط فسوف تضيع ذاكرة التشغيل وسياق الأدوات؛ ولو عولمت كسلسلة خطية فسوف تُهدر إمكانات التفريع والتوازي؛ ولو عولمت كوكيل حر بلا دستور فستزداد أخطاء التوسع والرسائل المكررة والأفعال غير المأذونة.

الوحدة الأساسية في M-TAOS ليست «الإجابة» بل **حلقة تشغيل ذات حالة** (Stateful Execution Loop). لكل حلقة: هدف، حالة أولية، رسم مشاكل، ميزانية حوسبة، ذاكرة قابلة للاسترجاع، مساحة أفعال، معايير نجاح، محاكم فشل، وأثر يُكتب من جديد في الحالة. بذلك تصبح المهمة أقرب إلى نظام تشغيل مصغر منه إلى استدعاء نموذج لغوي.

3. دستور المجال

3.1 الغرض

الغرض هو جعل كل مهمة مجدولة أو محادثة متكررة تزيد قدرة النظام الذي يستخدمها بمرور الوقت. المطلوب ليس أن تنتج كل ساعة نصاً جديداً فقط، بل أن تقلل مساحة المجهول، تبني أصولاً قابلة لإعادة الاستخدام، تنفذ أفعالاً حقيقية عند الإذن، وتترك حالة مرتبة تجعل التشغيل اللاحق يبدأ من حافة المعرفة السابقة لا من الصفر.

3.2 النطاق

يدخل في النطاق: بحث الويب، تحليل الملفات، إدارة الذاكرة، التحقق، التخطيط، توليد الأدوات، الاتصال بخدمات مثل Gmail وDrive عندما تكون مأذونة، التوليد البصري أو الوثائقي، اكتشاف الفرص، إدارة الحملات، تقييم النتائج، النسخ والإصدارات، والربط بين مهام متعددة. ويخرج من النطاق: التحايل على قيود المنصات، اختلاق قدرات حسابية غير موجودة، الادعاء بعمل خلفي غير مجدول، تضخيم حجم النص لذاته، إرسال جماعي غير منضبط، أو إخفاء عدم اليقين.

3.3 تعريف النجاح والفشل

النجاح يقاس بزيادة الفهم، الدقة الاسترجاعية، جودة القرار، عدد النتائج المتحققة، سرعة الوصول إلى الأثر، انخفاض التكرار، تراكم الذاكرة، ونسبة الفعل إلى الكلام. يفشل النظام عندما يتحول إلى مقال طويل، أو قائمة مصطلحات، أو تفريع بلا دمج، أو انعكاس ذاتي دائري، أو بحث بلا فعل، أو فعل بلا تحقق، أو ذاكرة خام لا يمكن استرجاعها، أو مخرجات ضخمة لا تغير أي قرار.

4. السؤال المركزي والتوتر والمتغير الخفي

السؤال المركزي: كيف نحول مهمة ChatGPT محدودة الزمن والسياق والأدوات إلى نظام حوسبة معرفية يتوسع تكيفياً، ينتج أكبر أثر موثوق في كل تشغيل، ويتراكم عبر التشغيلات دون أن ينهار في التكرار أو الهلوسة أو الكلفة؟

التوتر المركزي: التوسع مقابل الانضباط. مزيد من الفروع والأدوات والانعكاس يمكن أن يرفع الجودة، لكنه يمكن أيضاً أن يضاعف الأخطاء والكلفة. لذلك لا توجد قيمة في «حوسبة أكثر» إذا لم يكن هناك موجه للميزانية، ومتحقق، وقاعدة توقف.

المتغير الخفي: ليس حجم النموذج وحده، بل **جودة سياسة تخصيص الحوسبة**. الأوراق عن test-time scaling تشير إلى أن توزيع ميزانية الاستدلال بحسب صعوبة المسألة، التنويع، والتحقق يمكن أن يغير النتائج جذرياً [S06][S24][S25]. ويضيف M-TAOS أن تخصيص الحوسبة في المهام الوكيلية يجب أن يشمل أيضاً: اختيار الأدوات، البحث المتوازي، استرجاع الذاكرة، ترتيب الأفعال، وقرار متى نكتب الحالة إلى الذاكرة.

5. النواة التوليدية Generative Kernel

يمكن ضغط المجال في ثماني قواعد تعيد توليد بقية النظام:

1. **الغاية قبل النص:** لا تبدأ من «ما الذي سنكتب؟» بل من «ما التغير القابل للقياس المطلوب؟».
2. **الرسم قبل السلسلة:** حوّل المهمة إلى رسم مشكلات/اعتمادات، لأن أجزاء كثيرة يمكن تنفيذها بالتوازي بينما أجزاء أخرى تحتاج بيانات سابقة.
3. **الحوسبة مورد موجه:** وزع ميزانية التفكير والبحث والأدوات بحسب القيمة وعدم اليقين والمخاطر لا بالتساوي.
4. **الدليل سابق للادعاء:** كل عقدة عالية الأثر تحتاج طبقة إثبات أو وسم عدم يقين.
5. **الفعل جزء من الاستدلال:** الأداة ليست ملحقاً، نتيجة الفعل تغير حالة العالم وتصبح ملاحظة جديدة على طريقة ReAct [S01].
6. **الذاكرة انتقائية:** لا تحفظ كل شيء. استخلص القرارات، الأخطاء، الأدلة، الأصول، والأسئلة المفتوحة في طبقات قابلة للاسترجاع، مستفيداً من أفكار الذاكرة الهرمية [S08][S21].
7. **الهدم يسبق الإغلاق:** قبل اعتبار الفرع ناجحاً، ابحث عن سبب يجعله خاطئاً أو مكرراً أو ضعيف القيمة.
8. **كل تشغيل يرفع نقطة البداية:** الحالة النهائية ليست تقريراً فقط، بل مدخل منظم للتشغيل التالي.

هذه النواة تجعل M-TAOS نظاماً توليدياً: يمكنها إعادة بناء التخطيط، الذاكرة، التحقق، الأدوات، المقاييس، والمحاكم حتى لو أزيلت التفاصيل الثانوية.

6. المعجم التشغيلي الأساسي

المفهوم	التعريف	الوظيفة	خطر سوء الاستخدام	الرتبة
Task Kernel	الحد الأدنى من الغاية والقيود والحالة الذي يسمح بإعادة توليد خطة المهمة.	منع تضخم البرومبت مع الحفاظ على ما لا يجوز فقده.	إذا حذف قيداً حاكماً يصبح ضغطاً مدمراً.	تركيب نظري/هندسي
Execution Graph	رسم موجه لعقد العمل واعتماداتها وإمكان توازيها.	تحويل العمل من سلسلة إلى بنية قابلة للتجميع والتوازي.	الرسم المفرط قد يزيد كلفة التنسيق.	تركيب نظري/هندسي
Compute Budget	حصة الزمن/التوكنات/الأدوات/عمليات البحث المتاحة للتشغيل.	منع توهم حوسبة غير محدودة وتوجيه الموارد.	الميزانية الجامدة قد تهدر فرصاً عالية القيمة.	تركيب نظري/هندسي

المفهوم	التعريف	الوظيفة	خطر سوء الاستخدام	الرتبة
Vascular Compute Router	سياسة توزع الحوسبة على العقد بحسب القيمة وعدم اليقين والمركزية والمخاطر.	تخصيص الجهد حيث يحدث أكبر فرق.	قد يبالغ في استغلال تقدير قيمة خاطئ.	تركيب نظري/هندسي
HyperFrontier	مجموعة الفروع المفتوحة ذات أعلى منفعة متوقعة بعد آخر تشغيل.	استمرار العمل من حافة المعرفة بدلاً من إعادة البدء.	قد تتحول إلى backlog متضخم إن لم تنظف.	تركيب نظري/هندسي
Evidence Packet	حزمة ادعاء + مصدر + تاريخ + درجة ثقة + ما يثبته وما لا يثبته.	منع فصل النتيجة عن دليلها.	المصدر الموثوق لا يضمن صحة الاستنتاج.	تركيب نظري/هندسي
World-Effect Action	فعل يغير حالة خارجية: رسالة، ملف، سجل، تعديل، نشر، حجز، تحديث.	تحويل المعرفة إلى أثر.	يحتاج إذناً وحدوداً ومراقبة رجوع.	تركيب نظري/هندسي
Run Delta	ما تغير بين بداية التشغيل ونهايته: سجلات، أدلة، قرارات، أفعال، أخطاء.	قياس الإنتاج الحقيقي بدل طول الرد.	دلتا كبيرة يمكن أن تكون منخفضة الجودة.	تركيب نظري/هندسي
Self-Reconstruction Memory	ذاكرة تختزن ما يكفي لإعادة بناء سياق وقرارات التشغيل لاحقاً.	استمرارية طويلة المدى مع ضغط السياق.	قد ترسخ خطأً إذا لم تحفظ تعارضات الدليل.	تركيب نظري/هندسي
MEGA Grade	Multi-pass Evidence-Grounded Agency.	وصف تشغيل متعدد المرور: اكتشاف، تحقق، تركيب، هدم، فعل، تحديث.	ليس رتبة ذكاء ولا ضمان جودة.	تركيب نظري/هندسي
PETA Grade	Parallel Epistemic Tool-Action.	وصف تشغيل يوسع فروعاً مستقلة وأدوات متعددة مع دمج وتحكيم.	ليس مقياس bytes أو ادعاء petabyte.	تركيب نظري/هندسي
Destruction Court	بروتوكول هجومي يحاول إسقاط النتيجة قبل اعتمادها.	تقليل الثقة الزائفة والأصالة الوهمية.	المحكمة غير المحدودة قد تمنع الإغلاق.	تركيب نظري/هندسي
Compute Saturation	النقطة التي تصبح فيها منفعة حوسبة إضافية أقل من تكلفتها أو خطرها.	قاعدة توقف عقلانية.	تحتاج تقديراً تقريبياً للمنفعة.	تركيب نظري/هندسي
Cross-Task Diffusion	نقل أصل/قاعدة/قيد صالح من مهمة إلى مهمة أخرى دون نقل الضجيج.	زيادة الذكاء الجماعي لأسطول المهام.	خطر تلوث المجالات أو تسريب قيود غير مناسبة.	تركيب نظري/هندسي
Execution Proof	دليل ناتج عن تنفيذ أو تحقق مباشر، لا مجرد تبرير لغوي.	رفع وزن النتائج التي اجتازت اختباراً فعلياً.	قد تكون بيئة الاختبار نفسها ناقصة.	تركيب نظري/هندسي

7. الأساس البحثي: ماذا تضيف الأدبيات إلى Task OS؟

لا يساوي M-TAOS أي ورقة بعينها. بناؤه يعتمد على تجميع فروق متفرقة في الأدبيات. ReAct يبين قيمة التداخل بين الاستدلال والفعل [S01]. Tree of Thoughts وGraph of Thoughts يفتحان مساحة البحث على مسارات متعددة وبنى غير خطية [S02][S03]. Reflexion وSelf-Refine يقدمان آليات للاستفادة من التغذية الراجعة، لكن نتائج نقد التصحيح الذاتي تنبه إلى أن الانعكاس بلا إشارة خارجية قد يضر [S04][S05][S26]. MemGPT وCoALA يبرزان أن الذاكرة والسيطرة في الوكيل أقرب إلى هندسة نظام منها إلى مجرد سياق طويل [S08][S17]. LLMCompiler يبرهن أن بناء خطة اعتماد وتنفيذ استدعاءات مستقلة بالتوازي يمكن أن يقلل التأخير والتكلفة في بعض البيئات [S18].

أما مقاييس مثل AgentBench وGAIA وSWE-bench وOSWorld فتدفع نحو **التقييم التنفيذي**: لا يكفي أن يبدو الجواب مقنعاً، يجب أن ينجح في بيئة أو أداة أو اختبار [S13][S14][S15][S16]. GraphRAG وHippoRAG يضيفان درساً للذاكرة: الاسترجاع المتجهي المسطح لا يكفي للأسئلة العالمية أو متعددة القفزات، ويمكن أن تصبح العلاقات والبنية جزءاً من الذاكرة [S20][S21]. وأخيراً، test-time scaling ينقل السؤال من «ما النموذج الأكبر؟» إلى «كيف نوزع الحوسبة في وقت الحل؟» [S06][S24][S25].

7.1 محكمة الادعاء: الحوسبة الأكثر ليست دائماً أفضل

ينشأ خطأ شائع من تحويل نجاح بعض تقنيات test-time scaling إلى قاعدة مطلقة: «زد التفكير يتحسن الأداء». الأدبيات لا تدعم هذه الصيغة العامة. الفائدة تعتمد على صعوبة المسألة، بنية البحث، جودة المتحقق، التنويع، وسياسة الدمج [S06][S24]. كذلك توجد أدلة على أن التصحيح الذاتي الداخلي قد يرسخ الخطأ عندما لا يمتلك النموذج إشارة جديدة [S26]. لذا يعتمد M-TAOS مبدأ الحوسبة الحدّية: لا تُضاف جولة تفكير لأنها ممكنة، بل لأن لها مساراً واضحاً لإنتاج معلومة أو تحقق أو فعل لم يكن متاحاً في الجولة السابقة.

8. النظريات والقوانين الجديدة في M-TAOS

8.1 قانون المنفعة الحدية للحوسبة — Marginal Compute Value Law

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: تعريف القيمة الحدية لوحدة حوسبة إضافية بأنها مقدار التحسن المتوقع في الجودة/الأثر/انخفاض عدم اليقين مقسوماً على تكلفتها. الصيغة الإرشادية: MCV = Δ(U_verified)/ΔC. إذا هبطت MCV تحت عتبة المجال، يجب وقف الفرع أو تحويل الميزانية إلى فرع آخر.
الوظيفة في النظام: يمنع الانعكاس اللانهائي ويجعل «الدفع إلى الحد» متوافقاً مع الكفاءة.
بروتوكول اختبار أولي: اختبره بمقارنة سياسات ميزانية ثابتة مقابل ميزانية تكيفية على مهام حقيقية.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.2 نظرية الكثافة الإنجازية الحوسبية — Computational Action Density

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: قيمة التشغيل هي كثافة الدلتا المتحققة لكل حوسبة: سجلات صحيحة، أفعال ناجحة، قرارات محسنة، أصول قابلة لإعادة الاستخدام، وأسئلة مغلقة. طول الرد ليس متغيراً أساسياً.
الوظيفة في النظام: تحويل التحسين من «مزيد من النص» إلى «مزيد من الأثر لكل وحدة تشغيل».
بروتوكول اختبار أولي: قِس عدد الدلتا التي اجتازت التحقق بالنسبة للزمن/التوكنات/استدعاءات الأدوات.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.3 نظرية التوجيه الوعائي للحوسبة — Vascular Compute Routing

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: الحوسبة يجب أن تتدفق كشبكة وعائية نحو العقد التي تجمع قيمة متوقعة عالية، عدم يقين مفيد، مركزية اعتماد، وقابلية تحقق، مع خصم التكلفة والمخاطر.
الوظيفة في النظام: تمنع توزيع الجهد بالتساوي على فروع غير متساوية.
بروتوكول اختبار أولي: A/B بين موجه قيمة بسيط وتخصيص موحد.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.4 قانون حفظ الفرق الحاكم — Governing Difference Conservation

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: أي ضغط للسياق يجب أن يحافظ على الفروق التي تغير القرار: من فعل ماذا، ما الذي تحقق، ما الذي فشل، ما القيود، وما الفرع المفتوح.
الوظيفة في النظام: أساس ضغط المحادثات إلى ذاكرة دون قتل الديناميات.
بروتوكول اختبار أولي: اختبر قدرة وكيل جديد على إعادة بناء الخطة من الذاكرة المضغوطة.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.5 نظرية فرّع-اهدم-ادمج — Branch–Falsify–Merge Theory

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: التفريع وحده يولد تنوعاً لكن لا يضمن الحقيقة. يجب أن تمر الفروع عبر هدم مستقل ثم دمج يحافظ على الخلافات المبررة بدلاً من متوسط لغوي.
الوظيفة في النظام: مقاومة الانغلاق المبكر ومشكلة Degeneration of Thought.
بروتوكول اختبار أولي: قارن branch-only مع BFM على مسائل ذات بدائل متنافسة.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.6 قانون أسبقية الاختبار التنفيذي — Execution Proof Priority

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: عندما يتعارض تبرير لغوي مع نتيجة تنفيذية موثوقة، تعطى الأولوية للاختبار التنفيذي ضمن نطاق ما يقيسه.
الوظيفة في النظام: ينقل الوكيل من البلاغة إلى القابلية للاختبار.
بروتوكول اختبار أولي: Benchmark على مهام ملفات/كود/ويب ذات oracle تنفيذي.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.7 نظرية ذاكرة إعادة البناء — Reconstruction Memory Theory

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: الذاكرة الأفضل ليست الأكثر شمولاً بل التي تسمح باستعادة الحالة والقرارات والاعتمادات بأقل حجم ممكن وبأقل فقد حاكم.
الوظيفة في النظام: تربط ضغط السياق بالقدرة على الاستمرار.
بروتوكول اختبار أولي: قياس Kernel Regeneration Score بعد حذف السجل الخام.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.8 قانون انتشار المهارة عبر المهام — Cross-Task Skill Diffusion

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: إذا تكرر نمط نجاح أو فشل عبر ثلاث مهام مستقلة، يصبح مرشحاً للترقية من ملاحظة إلى HyperSkill عامة ثم يُختبر خارج مجال المصدر.
الوظيفة في النظام: تحويل أسطول المهام إلى بيئة تعلم جماعي.
بروتوكول اختبار أولي: اختبار انتقال skill مستخلصة إلى مهمة جديدة.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.9 نظرية تضاعف الأثر بالكتابة الفورية — Immediate Writeback Compounding

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: الدلتا غير المكتوبة تضيع أو تتكرر؛ أما الدلتا المكتوبة في سجل مركزي فتمنع إعادة العمل وتخلق فائدة مركبة في التشغيلات التالية.
الوظيفة في النظام: تبرير إلزامية Drive/Run Log في مهام الفرص.
بروتوكول اختبار أولي: قارن معدل التكرار قبل/بعد writeback.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.10 قانون تباعد الوكلاء النافع — Useful Agent Diversity

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: تعدد الوكلاء لا يفيد إذا أعادوا نفس المنظور. القيمة تأتي من اختلاف وظيفة المعرفة أو الدليل أو معيار الحكم، لا من تغيير الشخصية فقط.
الوظيفة في النظام: منع Multi-Agent theater.
بروتوكول اختبار أولي: قِس novelty overlap بين مخرجات الوكلاء مقابل أثرهم على القرار.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.11 نظرية حدود التوازي — Parallelism Boundary Theory

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: يمكن توازي العقد المستقلة فقط؛ أما العقد ذات الاعتماد أو المخاطر العالية فتحتاج نقاط مزامنة.
الوظيفة في النظام: منع تسريع خاطئ يؤدي إلى تناقضات أو أفعال متعارضة.
بروتوكول اختبار أولي: استخرج DAG واحسب critical path مقابل التنفيذ التسلسلي.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.12 قانون إغلاق العالم-الذاكرة — World-Memory Closure

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: لا تكتمل دورة الفعل حتى تتحول نتيجة العالم إلى ملاحظة ثم إلى تحديث في الذاكرة والسياسة.
الوظيفة في النظام: تحويل الأداة من فعل عابر إلى تعلم بين التشغيلات.
بروتوكول اختبار أولي: افحص نسبة الأفعال التي لها سجل نتيجة وتحديث لاحق.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.13 نظرية السطح المتعدد للمخرج — Multi-Surface Output Theory

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: النتيجة عالية القيمة يمكن أن تُجسّد في أكثر من سطح: chat، سجل، ملف، رسالة، خريطة، backlog. لكن يجب وجود مرجع مركزي يمنع الانقسام.
الوظيفة في النظام: زيادة قابلية الاستخدام دون خلق مصادر حقيقة متنافسة.
بروتوكول اختبار أولي: قياس divergence بين الأسطح مع canonical source.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.14 قانون الانفتاح الهجومي المنضبط — Disciplined Opening Law

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: فتح المجال يتطلب قلب المسلمات وربط المجالات واختبار الحالات القصوى، لكنه لا يُعتمد كمعرفة قبل عبور بوابات الفهم والآلية والدليل.
الوظيفة في النظام: إدماج Opening Intelligence دون فوضى نظرية.
بروتوكول اختبار أولي: تتبع نسبة الأفكار الجديدة التي تتحول إلى أدوات/اختبارات.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.15 نظرية الـHyperFrontier — Persistent Frontier Theory

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: يجب أن ينتهي التشغيل بقائمة فروع مفتوحة مرتبة بالقيمة، لا بخاتمة مغلقة. التشغيل التالي يبدأ من frontier بعد تحديث الأولويات.
الوظيفة في النظام: تراكم بحثي فعلي عبر الزمن.
بروتوكول اختبار أولي: قياس نسبة التشغيلات التي تبدأ من frontier لا من إعادة بحث مكرر.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.16 قانون مقاومة الضجيج — Noise Subtraction Law

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: كل زيادة في العدد لا تُعد تقدماً إذا رفعت التكرار أو الادعاءات غير المتحققة. ضمن MCD، الضجيج N يخصم من القيمة مهما كبر الحجم.
الوظيفة في النظام: حماية «mega output» من التحول إلى bulk غير مفيد.
بروتوكول اختبار أولي: مقارنة دقة/تكرار سجل قبل وبعد dedupe/court.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.17 نظرية الميزانية متعددة العملات — Multi-Currency Compute Budget

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: الحوسبة ليست tokens فقط: هناك زمن، بحث ويب، استدعاءات أدوات، سياق، انتباه، مخاطر، وحدود إرسال. الموجه الأمثل يوازن هذه العملات.
الوظيفة في النظام: منع تحسين بعد واحد على حساب النظام.
بروتوكول اختبار أولي: سجل cost vector لكل run وارتباطه بالدلتا.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

8.18 قانون التحقق غير المتناظر — Asymmetric Verification

الرتبة الإبستيمية: تركيب نظري/فرضية هندسية.
الصياغة: كلما ارتفع أثر الفعل أو صعوبة الرجوع عنه، يجب أن يرتفع مستوى التحقق قبل التنفيذ.
الوظيفة في النظام: دمج السلامة مع السرعة دون جعل كل شيء بطيئاً.
بروتوكول اختبار أولي: تعريف مستويات V0-V4 وربطها بحوادث الخطأ.
حد الفشل: لا يصبح القانون «حقيقة» بمجرد نجاحه في مهمة واحدة؛ يجب فصله عن أثر النموذج، جودة الأداة، نوع المجال، وطول السياق.

9. معادلات ومقاييس توجيه الحوسبة

المعادلات التالية إرشادية وليست مقاييس علمية معايرة بعد. هدفها إجبار المهندس على إظهار ما الذي يرفعه وما الذي يخصمه.

9.1 MCD للمهمة

```
MCD_task = λ1 D + λ2 I + λ3 T + λ4 A + λ5 P + λ6 C - λ7 N
```

حيث D كثافة الفروق المفيدة، I تكامل المستويات، T حساسية التحول/العتبات، A قوة الوكالة، P القدرة التنبؤية/الاختبارية، C الضغط القابل لإعادة البناء، وN الضجيج والتكرار والأصالة الزائفة.

9.2 دالة التوجيه الوعائي

```
Priority_i ∝ softmax(αV_i + βU_i + γK_i + δE_i - εCost_i - ζRisk_i)
```

V القيمة المتوقعة، U عدم اليقين القابل للاختزال، K مركزية العقدة في رسم الاعتماد، E قابلية إنتاج دليل/اختبار، Cost كلفة متعددة العملات، Risk أثر الخطأ. لا تعني softmax ضرورة تطبيق عددي؛ يمكن استعمالها كاستعارة انضباطية لترتيب الفروع.

9.3 كثافة الأثر

```
CAD = Verified_Delta / (Tokens + ToolCost + Latency + HumanAttention)
```

ويجب تفكيك Verified_Delta إلى أنواع: معلومة متحققة، سجل جديد، رسالة ناجحة، ملف مكتمل، اختبار ناجح، قرار مغلق، أو خطأ تم منعه.

9.4 معامل إعادة البناء

```
KRS = Ability_to_Reconstruct(State, Decisions, Frontier | Compressed_Memory)
```

يقاس تجريبياً بإعطاء وكيل جديد الذاكرة المضغوطة فقط ثم اختبار قدرته على متابعة المهمة دون إعادة اكتشاف واسع.

9.5 درجة MEGA/PETA

MEGA لا تُحسب بعدد الكلمات. الحد الأدنى لرتبة MEGA هو عبور خمس بوابات: Discovery, Verification, Synthesis, Falsification, Action/ Artifact. أما PETA فتضيف Parallel Branching, Tool Orchestration, Cross-Task Memory, and Merge Arbitration. أي تشغيل يفتقد التحقق أو الكتابة الراجعة لا يستحق الرتبة مهما كان حجمه.

10. خريطة المفاهيم

```
GOAL / WORLD CHANGE
        |
        v
[TASK KERNEL] --defines--> [SUCCESS TEST]
        |
        v
[PROBLEM GRAPH] --compiled to--> [EXECUTION DAG]
        |                           |
        |                           +--> parallel independent branches
        |                           +--> critical sequential path
        v
[VASCULAR COMPUTE ROUTER] <---- [BUDGET + RISK + UNCERTAINTY]
        |
        +--> [RETRIEVAL/MEMORY] ----> [EVIDENCE PACKETS]
        +--> [REASONING/SEARCH] -----> [CANDIDATE MODELS]
        +--> [TOOLS/ACTIONS] --------> [WORLD EFFECTS]
        +--> [CRITIQUE COURTS] ------> [REJECT/REPAIR]
        |
        v
[BRANCH-FALSIFY-MERGE COMPILER]
        |
        v
[VERIFIED RUN DELTA]
        |
        +--> chat/report/artifact
        +--> canonical state writeback
        +--> HyperSkill extraction
        +--> next HyperFrontier
        v
[SELF-RECONSTRUCTION MEMORY] --raises--> NEXT RUN STARTING INTELLIGENCE
```

11. خريطة المشكلات

المشكلة	الصياغة العميقة	السبب	إصلاح M-TAOS
Shallow Run	المهمة تنتج ملخصاً صغيراً رغم توفر وقت وأدوات وسياق.	غياب معيار للحد الأدنى من العمل أو frontier.	Compute Governor + workstream floor.
Serial Bottleneck	تنفيذ كل شيء واحداً بعد واحد رغم استقلال الفروع.	لا يوجد DAG أو موازاة.	Dependency compiler + parallel dispatch.
Reflection Loop	تكرار نقد ذاتي بلا معلومة جديدة.	الانعكاس غير مرتبط بدليل أو فعل.	MCV stop rule + external verifier.
Chat Amnesia	إعادة اكتشاف معلومات موجودة في محادثات سابقة.	السجل ليس ذاكرة منظمة.	episodic extraction + reconstruction memory.
Context Flood	إدخال كل التاريخ في السياق يرفع الضجيج.	لا توجد سياسة إدارة طبقات الذاكرة.	retrieval + kernelization + confidence tags.
Actionless Research	بحث واسع بلا رسالة/ملف/قرار/اختبار.	فصل المعرفة عن world-effect.	Action compiler + permission gate.
Duplicate Outreach	مهام متعددة تراسل الجهة نفسها.	غياب سجل مركزي وownership/ cooldown.	canonical ledger + dedupe key + suppression.

المشكلة	الصياغة العميقة	السبب	إصلاح M-TAOS
Bulk without Quality	الهدف العددي يدفع إلى سجلات ضعيفة أو مخترعة.	حافز حجم غير مقيد.	Noise penalty + verification thresholds.
False Novelty	إعادة تسمية مفهوم معروف على أنه نظرية جديدة.	لا توجد مقارنة prior art أو operational delta.	False Novelty Court.
Tool Misrouting	استدعاء أداة مكلفة أو غير مناسبة.	اختيار الأداة لا يدخل في التخطيط.	Tool Capability Graph + expected value routing.
Weak Merge	دمج فروع متعارضة بتسوية لغوية.	لا يحتفظ النظام بأسباب الاختلاف.	argument/evidence graph + adjudicator.
No Learning Between Runs	كل تشغيل مستقل.	لا writeback ولا error memory.	Run Delta + skill extraction + next frontier.
Unsafe High Agency	السرعة تسبق التحقق في أفعال خارجية.	لا تحقق غير متناظر.	V0-V4 verification levels + reversibility.
Artifact Drift	الملفات والسجلات تتباعد عن chat.	مصادر حقيقة متعددة.	canonical source + version graph.
Compute Theater	الادعاء بـ"max compute" دون عمل إضافي قابل للرصد.	قياس الأداء بطول الرد.	compute ledger + workstream/action counts.

12. خريطة الآليات

الآلية	المدخل	التحول	المخرج	فشل محتمل	اختبار
Kernelization	محادثة/طلب/سجل	استخراج الغاية والقيود والقرارات والفروق الحاكمة	Task Kernel	فقد قيد حاكم	اختبار إعادة البناء
DAG Compilation	قائمة أعمال	استخراج الاعتماد والتوازي ونقاط المزامنة	Execution Graph	اعتماد مخفي	إعادة تخطيط بعد كل observation حاسمة
Vascular Routing	رسم + budget	ترتيب الفروع بالقيمة/عدم اليقين/المخاطر	Allocation plan	تقدير قيمة منحاز	مراجعة دورية للpriority
Evidence Packetization	مصادر/نتائج أدوات	تطبيع claim/source/time/coverage/confidence	Evidence graph	مصدر قديم أو لا يثبت الادعاء	source/claim entailment check
Branch-Falsify-Merge	مرشحون متعددون	توليد مستقل ثم هدم ثم تحكيم	Surviving model/decision	تقارب مبكر	diversity floor + adversarial branch
World-Effect Loop	قرار مأذون	تنفيذ أداة ثم قراءة النتيجة	External delta	فعل غير قابل للرجوع	verification level + rollback if possible
Memory Consolidation	Run Delta	تصنيف episodic/semantic/procedural/suppression/frontier	Persistent memory	حفظ ضجيج	write filter + conflict log
Skill Distillation	نجاحات/أخطاء متكررة	استخراج invariant procedure + tests	HyperSkill	overgeneralization	cross-domain validation
Saturation Detection	مكاسب الجولات	تقدير marginal utility	Stop/redirect decision	وقف مبكر	minimum evidence floor
Cross-Task Diffusion	skill/constraint proven	تحديد applicability + scope	shared kernel update	context contamination	domain tags + opt-out

13. HyperSkills — قدرات قابلة للتدريب

13.1 HyperSkill: Kernel Extraction

التعريف: ضغط محادثة طويلة إلى غاية وقيود وقرارات وfrontier دون فقد الفرق الحاكم.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: سياق طويل أو متضارب.
بروتوكول التنفيذ: تمييز fact/decision/preference/open-loop؛ استخراج invariants؛ بناء page kernel-1؛ اختبار إعادة البناء.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: تلخيص يزيل الاستثناءات.
مؤشر الإتقان: KRS مرتفع.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

13.2 HyperSkill: Goal-to-World-Delta Compilation

التعريف: تحويل الهدف اللغوي إلى تغير خارجي ومعيار نجاح.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: أهداف فضفاضة مثل "نريد فرصاً أكثر".
بروتوكول التنفيذ: تعريف world state؛ delta؛ oracle؛ reversible actions؛ owner.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: مقاييس vanity.
مؤشر الإتقان: نجاح يمكن فحصه.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

13.3 HyperSkill: Dependency Graphing

التعريف: استخراج رسم اعتماد من خطة معقدة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: مهام متعددة الأدوات.
بروتوكول التنفيذ: تعريف nodes؛ edges؛ critical path؛ parallel sets؛ sync points.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: توازي عقد مترابطة.
مؤشر الإتقان: زمن أقل دون تناقض.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

13.4 HyperSkill: Adaptive Compute Routing

التعريف: تخصيص التفكير/البحث/الأدوات حسب القيمة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: ميزانية محدودة.
بروتوكول التنفيذ: تقدير V/U/K/E/Cost/Risk؛ allocate؛ reevaluate.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: الانحياز لفرع جذاب.
مؤشر الإتقان: مكاسب أعلى لكل كلفة.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Evidence Packet Engineering 13.5

التعريف: ربط كل ادعاء مهم بدليل وحدود.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: بحث ويب أو ملفات.
بروتوكول التنفيذ: claim normalization، source quality، scope; confidence; time.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: citation decoration.
مؤشر الإتقان: انخفاض claims unsupported.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Countermodel Generation 13.6

التعريف: إنشاء تفسير منافس حقيقي لا اعتراض شكلي.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: قرار أو نظرية.
بروتوكول التنفيذ: قلب الفرض، hidden variable، rival mechanism، discriminating evidence.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: strawman.
مؤشر الإتقان: فروع تغيّر القرار عند الدليل.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Branch Diversity Control 13.7

التعريف: توليد فروع مختلفة وظيفياً.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: بحث أو تخطيط.
بروتوكول التنفيذ: different evidence source; method; ontology; risk lens.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: paraphrase diversity.
مؤشر الإتقان: low semantic overlap + useful merge.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Falsification Routing 13.8

التعريف: توجيه الجهد نحو أضعف نقطة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: مرشح قوي ظاهرياً.
بروتوكول التنفيذ: identify crux; seek disconfirming data; edge cases.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: نقد عام.
مؤشر الإتقان: إسقاط مبكر للخيارات الرديئة.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Tool Capability Mapping 13.9

التعريف: معرفة ما الذي تستطيع كل أداة إثباته/تغييره.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: مجموعة أدوات.

بروتوكول التنفيذ: .schema; permissions; latency; evidence strength; reversibility
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: .tool fetishism
مؤشر الإتقان: استدعاءات أقل وأقوى.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Parallel Tool Orchestration 13.10

التعريف: تشغيل الأدوات المستقلة بالتوازي المنطقي.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: DAG.
بروتوكول التنفيذ: .batch queries; merge; conflict handling
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: .race/duplicate writes
مؤشر الإتقان: .latency lower; no duplicate action
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Episodic Memory Distillation 13.11

التعريف: تحويل سجل run إلى ذاكرة قابلة للاسترجاع.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: .chat/tool log
بروتوكول التنفيذ: .events; outcomes; errors; decisions; timestamps
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: حفظ كل شيء.
مؤشر الإتقان: استرجاع دقيق لمواقف مشابهة.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Semantic Memory Synthesis 13.12

التعريف: دمج نتائج متعددة إلى قواعد مستقرة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: episodes كثيرة.
بروتوكول التنفيذ: .cluster; contradict; confidence update; canonicalize
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: .premature generalization
مؤشر الإتقان: قواعد قابلة للنقل.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Procedural Skill Extraction 13.13

التعريف: استخراج SOP من نجاحات متكررة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: .successful runs
بروتوكول التنفيذ: .steps; preconditions; exceptions; tests; rollback
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.

خطأ شائع: copying accidental detail.
مؤشر الإتقان: نجاح في حالات جديدة.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Conflict-Preserving Merge 13.14

التعريف: دمج دون مسح الخلاف.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: فروع/مصادر متعارضة.
بروتوكول التنفيذ: map claims; evidence; boundary conditions; unresolved nodes.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: forced consensus.
مؤشر الإتقان: وضوح ما بقي مجهولاً.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Compute Saturation Detection 13.15

التعريف: تحديد متى لا يعود التفكير مربحاً.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: iterations.
بروتوكول التنفيذ: track new evidence/action per pass; marginal gain.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: stop by fatigue.
مؤشر الإتقان: lower cost at same verified quality.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Execution-Proof Design 13.16

التعريف: تصميم اختبار فعلي للنتيجة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: أداة/ملف/كود/ويب.
بروتوكول التنفيذ: oracle; pre/post state; assertions.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: proxy mismatch.
مؤشر الإتقان: functional success measurable.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: High-Agency Verification 13.17

التعريف: رفع التحقق مع الأثر.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: رسائل/تعديلات/نشر.
بروتوكول التنفيذ: recipient/source validation; cooldown; irreversible check.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: overchecking harmless actions.
مؤشر الإتقان: أخطاء خارجية أقل.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Canonical Ledger Engineering 13.18

التعريف: توحيد سجلات متعددة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: Drive/CRM/chat.
بروتوكول التنفيذ: stable IDs; dedupe keys; provenance; update not append.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: record fragmentation.
مؤشر الإتقان: duplicate rate near zero.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: HyperFrontier Management 13.19

التعريف: ترتيب ما لم ينجز بعد.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: open branches.
بروتوكول التنفيذ: EV; urgency; dependency; evidence readiness.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: backlog entropy.
مؤشر الإتقان: next run starts high-value work.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Cross-Task Knowledge Diffusion 13.20

التعريف: نقل قاعدة بين مهام.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: fleet logs.
بروتوكول التنفيذ: detect invariant; scope; inject; monitor regressions.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: negative transfer.
مؤشر الإتقان: fleet-wide improvement.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: False-Novelty Detection 13.21

التعريف: تمييز الابتكار عن إعادة التسمية.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: نظرية/أداة جديدة.
بروتوكول التنفيذ: prior-art analogs; operational delta; predictions.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: branding as novelty.
مؤشر الإتقان: novelty tied to new operation/test.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Research Attack Synthesis 13.22

التعريف: الهجوم على مجال ثم إعادة بنائه.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: مجموعة مصادر.

بروتوكول التنفيذ: .schools; contradictions; negative space; synthesis
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: .encyclopedic summary
مؤشر الإتقان: .new discriminating model
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Opportunity-to-Outreach Compiler 13.23

التعريف: تحويل lead إلى رسالة/فعل مضبوط.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: .B2B/job/research lead
بروتوكول التنفيذ: .verify; fit; role; prior contact; personalized angle; send/update
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: .bulk spam
مؤشر الإتقان: .reply/meeting yield
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Artifact Density Engineering 13.24

التعريف: ملء المستند بمعرفة لا فراغ.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: كتاب/تقرير.
بروتوكول التنفيذ: .hierarchy; tables only when comparative; callouts; no orphan headings
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: .visual density > cognitive density
مؤشر الإتقان: .high info/page + readability
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Source Freshness Routing 13.25

التعريف: تحديد متى يلزم الويب أو المصدر المتصل.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: ادعاءات زمنية.
بروتوكول التنفيذ: .freshness requirement; official source; recency
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: .stale knowledge
مؤشر الإتقان: .time-sensitive claims current
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Uncertainty Budgeting 13.26

التعريف: توزيع عدم اليقين لا إخفاؤه.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: ادعاءات مختلطة.
بروتوكول التنفيذ: .epistemic ranks; unresolved crux; next evidence
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.

خطأ شائع: false precision.
مؤشر الإتقان: better calibration.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Self-Correction with External Signal 13.27

التعريف: تصحيح مرتبط بمعلومة جديدة.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: draft/result.
بروتوكول التنفيذ: tool check; verifier; critique; revision.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: intrinsic loop.
مؤشر الإتقان: improvement causally linked to signal.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Multi-Agent Role Orthogonality 13.28

التعريف: تصميم وكلاء غير متكررين.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: agent team.
بروتوكول التنفيذ: assign evidence/reasoning/action/critique roles.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: persona theater.
مؤشر الإتقان: unique contribution per agent.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Run-Delta Compression 13.29

التعريف: إخراج فرق الحالة بدل سجل كامل.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: end of run.
بروتوكول التنفيذ: added/updated/failed/next; stable IDs.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: loss of rationale.
مؤشر الإتقان: compact but reconstructible.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Versioned Kernel Evolution 13.30

التعريف: تعديل kernel دون فقد history.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: OS prompts.
بروتوكول التنفيذ: semantic version; changelog; regression test.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: prompt drift.
مؤشر الإتقان: rollback possible.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Permission-Aware Agency 13.31

التعريف: استغلال الأذونات دون تجاوزها.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: connected tools.
بروتوكول التنفيذ: action class; standing authorization; sensitive boundary.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: under/over agency.
مؤشر الإتقان: correct actions without unnecessary confirmations.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Failure-Memory Encoding 13.32

التعريف: تحويل الخطأ إلى قيد وقائي.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: failed run.
بروتوكول التنفيذ: cause; trigger; signature; prevention; expiry.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: eternal stale ban.
مؤشر الإتقان: reduced recurrence.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Benchmark Construction 13.33

التعريف: تحويل هدف غامض إلى مجموعة اختبارات.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: OS capability.
بروتوكول التنفيذ: tasks; oracles; difficulty; ablations.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: teaching to test.
مؤشر الإتقان: generalization evidence.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: MCD Difference Scoring 13.34

التعريف: تقييم وحدة معرفة بحسب الفروق/التكامل/الأثر/الضجيج.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: concept/output.
بروتوكول التنفيذ: qualitative rubric; compare alternatives.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: fake precision.
مؤشر الإتقان: better ranking discipline.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: QMCD Model Superposition 13.35

التعريف: الإبقاء على نماذج متنافسة حتى ظهور دليل فاصِل.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: خلاف نظري.

بروتوكول التنفيذ: represent each model; boundary; collapse evidence.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: indecision.
مؤشر الإتقان: explicit discriminators.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: World-Effect Accounting 13.36

التعريف: تسجيل الأثر الفعلي.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: actions.
بروتوكول التنفيذ: pre/post; success; failure; downstream value.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: counting sends not outcomes.
مؤشر الإتقان: outcome-aware optimization.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Output Surface Compilation 13.37

التعريف: تحويل نتيجة واحدة إلى chat/ledger/file دون divergence.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: verified delta.
بروتوكول التنفيذ: canonicalize; render views; references.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: multiple truths.
مؤشر الإتقان: consistent views.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Mega-Grade Workstream Expansion 13.38

التعريف: فتح 3-10 مسارات قيمة في run.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: large task.
بروتوكول التنفيذ: decompose; prioritize; parallelize; merge.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: quantity theater.
مؤشر الإتقان: more verified deltas.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: PETA-Grade Runtime Arbitration 13.39

التعريف: التحكيم بين فروع/أدوات كثيرة تحت حدود runtime.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: large fleet/run.
بروتوكول التنفيذ: critical path; branch caps; stop rules; writes.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.

خطأ شائع: resource exhaustion.
مؤشر الإتقان: frontier reached efficiently.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

HyperSkill: Destruction-Court Operation 13.40

التعريف: تشغيل محاكم موجهة لا نقداً عاماً.
لماذا هي HyperSkill؟ لأنها لا تحل مثالاً واحداً، بل تغير طريقة توجيه المعرفة والحوسبة عبر مجالات متعددة.
المشكلة المستهدفة: major output.
بروتوكول التنفيذ: hallucination; reduction; novelty; operationalization; agency.
تمرين مبتدئ: طبق المهارة على تشغيل سابق واحد وحدد موضع قرار واضح قبل/بعد.
تمرين متقدم: طبقها على ثلاث مهام مختلفة ثم استخرج invariant لا يعتمد على المجال.
خطأ شائع: permanent skepticism.
مؤشر الإتقان: specific repairs.
صلة AGI: تدعم الانتقال من توليد نصي إلى تحكم ذاتي في الموارد، الذاكرة، الأدوات، والسياسات.

14. HyperLoops — حلقات التشغيل ذات الذاكرة

Kernel Extraction Loop 14.1

الهدف: تحويل raw context إلى kernel.
المدخل: raw chats/files/state.
المسار: extract differences → constraints → decisions → frontier.
المخرج: kernel + conflict log.
اختبار الجودة: reconstruction test.
نمط الفشل: overcompression.
الإصلاح: restore lost governing distinctions.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Evidence Ranking Loop 14.2

الهدف: رفع جودة الدليل.
المدخل: claims/sources.
المسار: source class → recency → entailment → confidence.
المخرج: evidence packets.
اختبار الجودة: unsupported claim rate.
نمط الفشل: citation theater.
الإصلاح: seek primary/official source.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Concept Disambiguation Loop 14.3

الهدف: منع الخلط المفاهيمي.
المدخل: terms.
المسار: define → contrast → boundary → test question.
المخرج: operational glossary.
اختبار الجودة: classification accuracy.
نمط الفشل: synonym collapse.
الإصلاح: add counterexamples.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Problem Decomposition Loop 14.4

الهدف: تحويل التعقيد إلى work graph.
المدخل: goal.
المسار: subproblems → dependencies → critical path → parallel sets.
المخرج: DAG.
اختبار الجودة: execution coverage.
نمط الفشل: fragmentation.
الإصلاح: merge by shared state.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Hypothesis Generation Loop 14.5

الهدف: فتح نماذج جديدة.
المدخل: gaps.
المسار: negative space → inversion → cross-domain → mechanism.
المخرج: hypotheses.
اختبار الجودة: novel predictions.
نمط الفشل: wordplay novelty.
الإصلاح: false novelty court.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Destruction and Repair Loop 14.6

الهدف: هدم أفضل مرشح.
المدخل: candidate.
المسار: crux → counterevidence → edge cases → repair.
المخرج: survivor v2.
اختبار الجودة: resistance score.
نمط الفشل: defensive rationalization.
الإصلاح: independent critic.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Tool Construction Loop 14.7

الهدف: تحويل حاجة متكررة إلى tool.
المدخل: repeated manual pattern.
المسار: inputs → algorithm → output → test → misuse.
المخرج: tool spec.
اختبار الجودة: time/error reduction.
نمط الفشل: overtooling.
الإصلاح: minimum viable tool.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Skill Training Loop 14.8

الهدف: تحويل tool/procedure إلى capability.
المدخل: procedure.
المسار: exercise → feedback → variation → mastery test.
المخرج: HyperSkill.
اختبار الجودة: transfer success.

نمط الفشل: rote execution.
الإصلاح: cross-domain test.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Retrieval Optimization Loop 14.9

الهدف: تحسين ما يُسترجع.
المدخل: query + corpus.
المسار: route query type → retrieve → grade → expand/graph.
المخرج: context packet.
اختبار الجودة: precision/recall proxy.
نمط الفشل: context flood.
الإصلاح: rerank + chunk policy.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Agentic Execution Loop 14.10

الهدف: إغلاق goal-action-observation.
المدخل: plan.
المسار: act → observe → update → replan.
المخرج: world delta.
اختبار الجودة: functional success.
نمط الفشل: open-loop action.
الإصلاح: state readback.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Update and Versioning Loop 14.11

الهدف: منع drift.
المدخل: kernel/state.
المسار: diff → regression → version → migrate.
المخرج: new version.
اختبار الجودة: rollback readiness.
نمط الفشل: silent prompt mutation.
الإصلاح: changelog + test.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Research Program Loop 14.12

الهدف: تحويل gaps إلى agenda.
المدخل: unknowns.
المسار: rank → hypothesis → method → evidence → decision.
المخرج: research backlog.
اختبار الجودة: closed questions/run.
نمط الفشل: endless reading.
الإصلاح: world-effect milestone.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Compute Routing Loop 14.13

الهدف: إعادة توزيع budget.
المدخل: branch metrics.
المسار: measure MCV → reallocate → stop/expand.

المخرج: budget plan.
اختبار الجودة: utility per cost.
نمط الفشل: equal allocation.
الإصلاح: vascular prioritization.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Parallel Branch Loop 14.14

الهدف: استغلال الاستقلال.
المدخل: DAG.
المسار: spawn → tool/search → local verify → sync.
المخرج: branch packets.
اختبار الجودة: latency reduction.
نمط الفشل: race duplication.
الإصلاح: dedupe/sync barrier.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Branch-Falsify-Merge Loop 14.15

الهدف: دمج مضبوط.
المدخل: branch packets.
المسار: adversarial test → evidence graph → boundary merge.
المخرج: decision/model.
اختبار الجودة: calibration.
نمط الفشل: forced consensus.
الإصلاح: retain unresolved conflicts.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

World-Effect Loop 14.16

الهدف: تحويل المعرفة لأثر.
المدخل: verified action candidate.
المسار: permission → precheck → execute → readback.
المخرج: external state change.
اختبار الجودة: outcome.
نمط الفشل: unsafe send/write.
الإصلاح: asymmetric verification.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Memory Consolidation Loop 14.17

الهدف: تثبيت التعلم.
المدخل: run delta.
المسار: filter → classify → dedupe → conflict → store.
المخرج: memory update.
اختبار الجودة: next-run reuse.
نمط الفشل: memory dumping.
الإصلاح: write policy.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Failure Reconstruction Loop 14.18

الهدف: تعلم من الخطأ.
المدخل: failure.
المسار: timeline → root cause → missed signal → guardrail.
المخرج: failure memory.
اختبار الجودة: recurrence reduction.
نمط الفشل: blame narrative.
الإصلاح: counterfactual replay.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Cross-Task Diffusion Loop 14.19

الهدف: رفع fleet intelligence.
المدخل: validated skill.
المسار: scope → candidate tasks → inject → monitor.
المخرج: shared kernel delta.
اختبار الجودة: fleet delta.
نمط الفشل: negative transfer.
الإصلاح: domain tags/rollback.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Opportunity Conversion Loop 14.20

الهدف: من lead إلى outcome.
المدخل: verified opportunity.
المسار: fit → contact → message → send → follow-up → classify.
المخرج: pipeline state.
اختبار الجودة: reply/meeting/job/revenue.
نمط الفشل: send counting.
الإصلاح: outcome accounting.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Artifact Production Loop 14.21

الهدف: تحويل knowledge إلى file.
المدخل: structured content.
المسار: layout → render → inspect → repair → deliver.
المخرج: verified artifact.
اختبار الجودة: visual/readability checks.
نمط الفشل: unverified export.
الإصلاح: render inspection.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Hallucination Suppression Loop 14.22

الهدف: منع claims بلا سند.
المدخل: draft.
المسار: extract claims → classify → source/check → revise.
المخرج: governed draft.
اختبار الجودة: unsupported rate.

نمط الفشل: over-citation.
الإصلاح: claim-level governance.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

Saturation Loop 14.23

الهدف: وقف الحوسبة عند الحد المفيد.
المدخل: iterations.
المسار: track novelty/evidence/action → MCV threshold.
المخرج: stop/redirect.
اختبار الجودة: same quality lower cost.
نمط الفشل: premature stop.
الإصلاح: minimum-pass guarantee.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

HyperFrontier Loop 14.24

الهدف: استمرار بين التشغيلات.
المدخل: open queue.
المسار: reprioritize by state → execute top → add new frontier.
المخرج: next queue.
اختبار الجودة: progress continuity.
نمط الفشل: stale backlog.
الإصلاح: expiry + pruning.
تحديث الذاكرة: سجل الدلتا، القرار، سبب الرفض/القبول، وما الذي يجب أن يتغير في التشغيل التالي.

15. طبقة الوكلاء Agentic Layer

الوكيل	الدور	المدخل	المخرج	ممنوع	التقييم
Kernel Agent	يستعيد ويضغط الحالة	chat/files/ledger	Task Kernel + conflicts	لا يختلق حقائق مفقودة	KRS
Research Scout	يوسع فضاء المصادر/الفرص	query/frontier	candidate evidence	لا يعتمد مرشحاً نهائياً	coverage + freshness
Evidence Agent	يفحص claim-source relation	claims/sources	evidence packets	لا يساوي وجود رابط بالإثبات	support precision
Graph Compiler	يبني /concept/problem task DAG	kernel/candidates	graphs	لا يفرض توازي عقد مترابطة	dependency correctness
Compute Governor	يوزع الميزانية ويوقف التشبع	graph/budget/metrics	allocation decisions	لا يقيس النجاح بطول النص	utility/cost
Parallel Executor	ينفذ الفروع المستقلة	parallel set/tools	branch results	لا يكرر نفس action	latency + dedupe
Critique Agent	يبحث عن crux والفشل	candidate output	attack packet	لا يكتفي باعتراضات عامة	rejection/repair value
Adjudicator	يدمج خلافات الأدلة	branch + attack packets	decision + uncertainty	لا يمحو الخلاف	calibration
Tool Agent	يختار/يبني الأدوات	task node/capability graph	tool plan/result	لا يستخدم أداة خارج الصلاحية	functional success
Memory Agent	يكتب ويسترجع الذاكرة	run delta/store	memory packets	لا يحفظ secrets بلا حاجة	retrieval utility

الوكيل	الدور	المدخل	المخرج	ممنوع	التقييم
Skill Distiller	يستخرج procedures العامة	run histories	HyperSkills	لا يعمم من حالة واحدة	transfer score
Opportunity Agent	يحوّل lead إلى pipeline	lead/ledger/Gmail	outreach/action	لا يخمن emails	reply/outcome rate
Artifact Agent	ينتج الملفات ويتحقق بصرياً	content/spec	PDF/MD/etc	لا يسلم artifact غير مفحوص	render quality
Safety/Permission Agent	يضبط الأفعال الحساسة	candidate /action context	allow/draft/block	لا يوسع التفويض ضمنياً	incident prevention
Versioning Agent	يدير kernel/config evolution	current/new state	version + changelog	لا يستبدل بلا rollback	regression pass
Benchmark Agent	يبني اختبارات واقعية	capability target	benchmark + oracle	لا يصمم test سهل لإثبات النظرية	discriminative power
Fleet Diffusion Agent	ينقل skill بين tasks	fleet states	injection candidates	لا يلوث domains	cross-task gain
Court Destruction Agent	يشغل المحاكم الخمس	major output	verdict + repairs	لا يتحول إلى veto دائم	destruction resistance

16. طبقة الأدوات Tool Layer

16.1 Concept Kernelizer

الغرض: استخراج النواة من corpus/chat.
المدخلات: raw text + state.
الخوارزمية/الآلية: compression → invariant selection → difference extraction.
المخرجات: kernel + lost-info warnings.
التقييم: reconstruction test.
مخاطر سوء الاستخدام: حذف الاستثناءات.
طريقة التحديث: versioned templates.

16.2 Evidence Ranker

الغرض: ترتيب الأدلة.
المدخلات: claim-source packets.
الخوارزمية/الآلية: authority/recency/directness/independence.
المخرجات: ranked evidence.
التقييم: human/benchmark agreement.
مخاطر سوء الاستخدام: تزيين الاستشهاد.
طريقة التحديث: domain calibrations.

16.3 Difference Extractor

الغرض: كشف الفروق الحاكمة.
المدخلات: models/cases.
الخوارزمية/الآلية: contrast dimensions → decision sensitivity.
المخرجات: difference map.
التقييم: decision impact.
مخاطر سوء الاستخدام: trivial differences.
طريقة التحديث: learn from court outcomes.

MCD Evaluator 16.4

الغرض: تقييم معرفي نوعي.
المدخلات: concept/output.
الخوارزمية/الآلية: D/I/T/A/P/C/N rubric.
المخرجات: profile.
التقييم: expert consistency.
مخاطر سوء الاستخدام: fake scalar precision.
طريقة التحديث: keep qualitative bins.

QMCD Superposition Map 16.5

الغرض: إبقاء النماذج المتنافسة.
المدخلات: candidate models.
الخوارزمية/الآلية: claims/boundaries/discriminators.
المخرجات: superposition graph.
التقييم: collapse by evidence.
مخاطر سوء الاستخدام: permanent ambiguity.
طريقة التحديث: expiry + experiment.

Task DAG Compiler 16.6

الغرض: بناء رسم التنفيذ.
المدخلات: goal/workstreams.
الخوارزمية/الآلية: dependency parse → critical path.
المخرجات: DAG.
التقييم: execution trace.
مخاطر سوء الاستخدام: hidden deps.
طريقة التحديث: replan from observations.

Vascular Compute Router 16.7

الغرض: تخصيص الميزانية.
المدخلات: DAG + branch values.
الخوارزمية/الآلية: priority function + MCV.
المخرجات: allocation.
التقييم: utility per cost.
مخاطر سوء الاستخدام: wrong value estimate.
طريقة التحديث: online update.

Parallel Dispatch Planner 16.8

الغرض: تجميع الاستدعاءات المستقلة.
المدخلات: DAG/tools.
الخوارزمية/الآلية: batch/parallel groups.
المخرجات: dispatch plan.
التقييم: latency.
مخاطر سوء الاستخدام: rate-limit collision.
طريقة التحديث: tool telemetry.

Branch Diversity Meter 16.9

الغرض: قياس استقلال الفروع.
المدخلات: branch drafts.
الخوارزمية/الآلية: semantic/function/evidence overlap.
المخرجات: diversity report.
التقييم: marginal contribution.
مخاطر سوء الاستخدام: cosmetic diversity.
طريقة التحديث: role redesign.

Destruction Court Matrix 16.10

الغرض: هجومية منظمة.
المدخلات: major output.
الخوارزمية/الآلية: five courts + crux severity.
المخرجات: verdicts.
التقييم: error catches.
مخاطر سوء الاستخدام: ritual criticism.
طريقة التحديث: learn failure signatures.

HyperSkill Distiller 16.11

الغرض: استخراج skill.
المدخلات: episodes.
الخوارزمية/الآلية: invariant SOP + preconditions + tests.
المخرجات: skill card.
التقييم: transfer.
مخاطر سوء الاستخدام: overfit.
طريقة التحديث: cross-domain trials.

HyperLoop Orchestrator 16.12

الغرض: تشغيل حلقات مع state.
المدخلات: task state.
الخوارزمية/الآلية: loop selection + transitions.
المخرجات: trace.
التقييم: goal completion.
مخاطر سوء الاستخدام: loop explosion.
طريقة التحديث: saturation controls.

RAG Chunking Compiler 16.13

الغرض: إعداد corpus.
المدخلات: documents.
الخوارزمية/الآلية: chunk type + metadata + boundaries.
المخرجات: index units.
التقييم: retrieval precision.
مخاطر سوء الاستخدام: arbitrary chunks.
طريقة التحديث: query telemetry.

GraphRAG Relation Compiler 16.14

الغرض: استخراج العلاقات.
المدخلات: chunks.
الخوارزمية/الآلية: nodes/edges/community summaries.
المخرجات: knowledge graph.
التقييم: global/local QA.
مخاطر سوء الاستخدام: false edges.
طريقة التحديث: confidence edges.

Memory Write Filter 16.15

الغرض: تحديد ما يحفظ.
المدخلات: run delta.
الخوارزمية/الآلية: value/reuse/conflict/privacy.
المخرجات: memory update.
التقييم: future retrieval utility.
مخاطر سوء الاستخدام: memory dumping.
طريقة التحديث: decay/forgetting.

Failure Signature Detector 16.16

الغرض: تعرف أخطاء متكررة.
المدخلات: logs.
الخوارزمية/الآلية: cluster triggers/root causes.
المخرجات: guardrail candidates.
التقييم: recurrence.
مخاطر سوء الاستخدام: false pattern.
طريقة التحديث: minimum support count.

Execution Proof Harness 16.17

الغرض: اختبار مخرجات.
المدخلات: artifact/action/code.
الخوارزمية/الآلية: oracle/pre-post assertions.
المخرجات: pass/fail evidence.
التقييم: functional correctness.
مخاطر سوء الاستخدام: bad oracle.
طريقة التحديث: version tests.

Outreach Dedupe Gate 16.18

الغرض: منع تكرار الرسائل.
المدخلات: ledger/Gmail/lead.
الخوارزمية/الآلية: stable id/domain/recipient/cooldown.
المخرجات: allow/suppress.
التقييم: duplicate rate.
مخاطر سوء الاستخدام: false suppress.
طريقة التحديث: material-change override.

Run Delta Compiler 16.19

الغرض: تلخيص التغير.
المدخلات: before/after state.
الخوارزمية/الآلية: diff by record/action/evidence/frontier.
المخرجات: delta packet.
التقييم: reconstruction.
مخاطر سوء الاستخدام: narrative logs.
طريقة التحديث: typed schema.

HyperFrontier Ranker 16.20

الغرض: ترتيب التالي.
المدخلات: open nodes.
الخوارزمية/الآلية: EV/urgency/dependency/readiness.
المخرجات: queue.
التقييم: throughput.
مخاطر سوء الاستخدام: stale priority.
طريقة التحديث: decay/re-score.

AGI-Readiness Index 16.21

الغرض: تقييم OS.
المدخلات: metrics.
الخوارزمية/الآلية: qualitative multi-axis profile.
المخرجات: readiness report.
التقييم: benchmark correlations.
مخاطر سوء الاستخدام: single score fetish.
طريقة التحديث: retain vector.

Prompt/Kernel Regression Tester 16.22

الغرض: اختبار تحديثات task prompts.
المدخلات: old/new kernel + cases.
الخوارزمية/الآلية: replay representative cases.
المخرجات: regression diff.
التقييم: no critical loss.
مخاطر سوء الاستخدام: overfit suite.
طريقة التحديث: expand from incidents.

Cross-Task Diffusion Compiler 16.23

الغرض: تحديث fleet.
المدخلات: validated kernel delta.
الخوارزمية/الآلية: scope mapping + injection.
المخرجات: task patches.
التقييم: fleet uplift.
مخاطر سوء الاستخدام: negative transfer.
طريقة التحديث: rollback.

Compute Ledger 16.24

الغرض: تسجيل ما استُهلك وما نتج.
المدخلات: run telemetry.
الخوارزمية/الآلية: tokens/time/tools/actions/deltas.
المخرجات: efficiency profile.
التقييم: CAD/MCV trends.
مخاطر سوء الاستخدام: missing telemetry.
طريقة التحديث: approximation with confidence.

17. هندسة الذاكرة وRAG/GraphRAG

يقترح M-TAOS ذاكرة بخمس طبقات، مستفيداً مفاهيمياً من الذاكرة الهرمية في MemGPT ومن الأطر المعرفية مثل CoALA ومن استرجاع العلاقات في GraphRAG/HippoRAG [S08][S17][S20][S21]. الطبقات ليست مجرد مخازن مختلفة؛ لكل طبقة سياسة كتابة واسترجاع مختلفة:

1. **Working Context:** ما يلزم للخطوة الحالية فقط.
2. **Episodic Run Memory:** أحداث التشغيل ونتائج الأدوات والأخطاء والرسائل والردود مع الزمن.
3. **Semantic Canonical Memory:** حقائق/قواعد مدمجة مع سجل التعارض والثقة.
4. **Procedural HyperSkill Memory:** إجراءات مجربة وشروطها واختباراتها.
5. **Frontier & Governance Memory:** القيود، suppression، الأذونات، إصدارات kernel، والفروع المفتوحة.

قانون الكتابة: لا يُكتب العنصر لأنه ظهر؛ يُكتب إذا كان سيغير قراراً لاحقاً، يمنع تكراراً، يدعم إعادة بناء، أو يمثل التزاماً/حالة خارجية. وقانون الاسترجاع: نوع السؤال يحدد مسار الاسترجاع. سؤال «ماذا حدث مع هذا العميل؟» يذهب إلى episodic؛ سؤال «ما أفضل قاعدة إرسال؟» إلى procedural/semantic؛ سؤال «ما الذي لم نختبره؟» إلى frontier.

Chunk types 17.1

- concept chunk
- problem chunk
- mechanism chunk
- evidence packet
- decision chunk
- tool/action chunk
- failure chunk
- skill chunk
- loop chunk
- agent contract chunk
- frontier chunk
- version chunk

Metadata الدنيا 17.2

```
domain, task_id, run_id, timestamp, entity, claim_type, confidence, provenance, centrality,
actionability, risk, version, expiry, suppression, relation_types
```

Graph nodes/edges 17.3

العقد: Goal, Claim, Source, Entity, Problem, Mechanism, Action, Tool, Skill, Failure, Agent, Test, Version. الحواف: supports, contradicts, depends_on, blocks, generates, operationalizes, verifies, falsifies, updates, contacted, replied, supersedes, transfers_to.

17.4 الاسترجاع المحلي والعالمي

الاسترجاع المحلي مناسب لسؤال عن جهة أو سجل محدد. أما الأسئلة العالمية مثل «ما أكثر القطاعات استجابة؟» أو «ما نمط الفشل عبر المهام؟» فتحتاج تجميعاً مجتمعياً/بيانياً يشبه الدافع وراء GraphRAG: لا يكفي nearest-neighbor على chunk واحد [S20].

18. مترجم التنفيذ Runtime Frontier Compiler

مترجم M-TAOS يحول المهمة قبل التنفيذ إلى تمثيل وسيط شبيه بخطة compiler. الفكرة ليست مساواة اللغة بالبرمجة، بل الاستفادة من درس LLMCompiler: عندما تُعرف الاعتمادات يمكن تجنب التسلسل غير الضروري [S18].

المرحلة **A — Parse**: استخرج الغاية، القيود، الأفعال المسموحة، المخرجات، وحالة البداية.
المرحلة **B — Graph**: ابن عقداً من نوع Research, Retrieve, Verify, Generate, Act, Writeback, Court.
المرحلة **C — Analyze**: حدد critical path، dependencies، فرص التوازي، وحدود rate/permission.
المرحلة **D — Allocate**: طبق Vascular Compute Router.
المرحلة **E — Execute**: نفذ مجموعات مستقلة، ثم اقرأ نتائج العالم.
المرحلة **F — Replan**: أي ملاحظة تغير الفرض أو صلاحية فرع تعيد رسم الجزء المتأثر.
المرحلة **G — Merge**: Branch-Falsify-Merge.
المرحلة **H — Commit**: اكتب Run Delta والذاكرة والـfrontier.

هذا المترجم هو قلب السرعة: «fast» لا يعني تقليل التفكير، بل تقليل الانتظار التسلسلي والتكرار وإعادة البحث. و«huge» لا يعني تضخيم كل فرع، بل فتح فروع مستقلة كثيرة بحدود ودمج مضبوط.

19. مستويات MEGA/PETA للتشغيل

المستوى	الحد الأدنى	ما لا يكفي للوصول إليه
G0 — Reply	إجابة صحيحة محلية	طول النص
G1 — Structured	تفكيك + مخرج منظم	عناوين كثيرة
G2 — Evidence	مصادر/تحقق + حدود	روابط بلا claim mapping
G3 — Agentic	أداة/فعل + observation + replanning	خطة بلا تنفيذ
MEGA	≥3 + workstreams + multi-pass + evidence court + artifact/action + writeback	«تحليل عميق» لغوي فقط
PETA	MEGA + parallel branches/tools + dependency graph + cross-task memory + arbitration + frontier	آلاف البنود غير المتحققة

يمكن لتشغيل قصير أن يكون MEGA إذا أغلق حلقات كثيرة عالية القيمة. ويمكن لتقرير 100 صفحة أن يبقى G1 إذا لم يمر بتحقق أو لا يغير حالة.

20. محاكم الهدم

20.1 محكمة الهلوسة

الأسئلة الهجومية: هل يوجد ادعاء بلا مصدر/اختبار؟ هل المصدر يثبت ما نُسب إليه؟ هل خلطنا الاستنتاج بالحقيقة؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: إزالة/خفض رتبة/طلب تحقق.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلاً قابلاً للرصد أو قرار إغلاق.

20.2 محكمة الاختزال

الأسئلة الهجومية: هل ضغطنا السياق حتى فقدنا استثناءً أو تعارضاً أو علاقة زمنية؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: إعادة الفرق الحاكم إلى kernel.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.3 محكمة الأصالة الزائفة

الأسئلة الهجومية: هل الاسم الجديد يضيف عملية أو تنبؤاً أم يعيد تسمية مفهوم معروف؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: ربط novelty بـ operational delta أو حذف دعوى الجدة.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.4 محكمة ضعف التشغيل

الأسئلة الهجومية: هل يمكن تشغيل المفهوم؟ ما مدخلاته ومخرجاته واختباره؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: تحويله إلى tool/skill/loop أو إسقاطه.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.5 محكمة فشل الوكيل

الأسئلة الهجومية: لو أعطينا هذا الKB لوكيل، أين سيتخذ قراراً أسوأ؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: guardrails + forbidden ops + escalation.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.6 محكمة تشبع الحوسبة

الأسئلة الهجومية: هل الجولة الإضافية أنتجت دليلاً أو فعلًا جديداً؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: stop/redirect.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.7 محكمة التكرار

الأسئلة الهجومية: هل يعيد هذا الفرع/السجل شيئاً موجوداً؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: dedupe/merge.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.8 محكمة الأثر

الأسئلة الهجومية: هل نعد النشاط إنجازاً رغم عدم وجود world delta؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: إعادة تعريف success metric.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.9 محكمة التوازي

الأسئلة الهجومية: هل شغلنا عقداً بالتوازي رغم اعتماد خفي؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: إضافة sync/dependency.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

20.10 محكمة الذاكرة

الأسئلة الهجومية: هل كتبنا ما لا يفيد أو أهملنا قراراً يجب حفظه؟
الحكم الممكن: PASS / REPAIR / REJECT / NEEDS-VERIFICATION.
الإصلاح: memory filter repair.
لا يجوز للمحكمة أن تصبح طقساً إنشائياً؛ يجب أن تنتج تعديلًا قابلاً للرصد أو قرار إغلاق.

21. طبقة التقييم وAGI-Readiness

المقياس	سؤال القياس	ملاحظة
Kernel Regeneration Score	هل يستطيع وكيل جديد متابعة المهمة من kernel فقط؟	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Concept Operationality	نسبة المفاهيم التي لها inputs/mechanism/test.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Evidence Governance	نسبة الادعاءات عالية الأثر ذات provenance/ confidence.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Retrieval Precision	مقدار السياق المسترجع الذي يؤثر فعلاً في القرار.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Graph Connectivity	هل الروابط تمثل الاعتمادات/التعارضات الحقيقية؟	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
HyperSkill Density	عدد الإجراءات القابلة للنقل نسبة إلى حجم المعرفة.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
HyperLoop Maturity	وجود input/output/test/memory لكل loop.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Destruction Resistance	عدد الاعتراضات القاتلة التي صمد أمامها بعد repair.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Agentic Deployment	قدرة النظام على تنفيذ أفعال مأذونة مع readback.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Research Opening	عدد الأسئلة/الفرضيات الجديدة ذات مميز تجريبي.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Update Capacity	سرعة دمج دلتا جديدة دون كسر السابق.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
MCD Profile	D/I/T/A/P/C/N نوعياً.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Computational Action Density	Verified Delta لكل كلفة متعددة العملات.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Duplicate Avoidance	نسبة ما تم منعه من العمل/الإرسال المكرر.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
World-Effect Yield	الأفعال التي حققت نتيجة لاحقة ذات قيمة.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Frontier Continuity	نسبة التشغيلات التي تبدأ من frontier صحيح.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Cross-Task Transfer	تحسن مهام أخرى بعد نقل HyperSkill.	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.
Compute Saturation Accuracy	هل توقف النظام قرب النقطة ذات المنفعة الحدية المنخفضة؟	يفضل ملف نوعي/فئات إلى أن تتوفر telemetry صالحة للقياس العددي.

22. بروتوكول تشغيل مهمة واحدة

المرحلة 0 — State Load

استرجع ما يكفي من المحادثة، السجل، الملفات، والـfrontier. لا تعيد سؤالاً سبق حسمه. صنّف كل معلومة إلى fact/constraint/decision/open-loop.

المرحلة 1 — World Delta

اكتب في سطر واحد: «إذا نجح هذا التشغيل، ما الذي سيتغير خارج النص؟». إذا لم يوجد أثر خارجي، عرّف artifact أو قراراً أو knowledge delta قابلاً للفحص.

المرحلة 2 — Graph Compile

افتح 3–10 workstreams عندما يسمح المجال. كل workstream يجب أن يكون مختلفاً وظيفياً: مصدر مختلف، مشكلة فرعية، أداة، تحقق، أو مسار فعل.

المرحلة 3 — Compute Allocation

حدد critical path والفرع عالي عدم اليقين والفرع عالي الأثر. وزع الجهد بشكل غير متساو. لا تحرق معظم الميزانية على مقدمة يمكن ضغطها.

المرحلة 4 — Discovery/Execution

نفذ الاسترجاع والبحث والأدوات. اجعل النتائج المتوازية مستقلة. سجّل evidence packets لا مجرد ملاحظات.

المرحلة 5 — Verification

للادعاءات: source/claim check. للأفعال: pre/post check. للملفات: render/test. للفرص: entity/contact/dedupe/cooldown. للكود: tests. للنظريات: prior-art/operational delta.

المرحلة 6 — Branch-Falsify-Merge

اهدم أفضل فروعك. إذا بقي خلاف، احتفظ به مع discriminator بدلاً من إخفائه. إذا سقط فرع، سجل سبب سقوطه كي لا يعود التشغيل التالي لاكتشافه.

المرحلة 7 — World Effect

نفذ الأفعال المأذونة ذات أعلى قيمة. اقرأ نتيجة الأداة. لا تعتبر «تم الإرسال» نهاية إذا كان الهدف اجتماعاً أو وظيفة؛ يصبح الإرسال حالة وسيطة في pipeline.

المرحلة 8 — Commit

اكتب Run Delta، حدّث المصدر المركزي، استخرج skills/failures، وأنشئ HyperFrontier جديداً.

المرحلة 9 — Saturation Court

اسأل: هل جولة إضافية ستغير قراراً أو تضيف دليلاً مستقلاً أو تنفذ فعلاً؟ إن لم يكن، أغلق التشغيل بدلاً من إنتاج حشو.

23. بروتوكول أسطول المهام Task Fleet OS

عند وجود مهام كثيرة، تصبح الوحدة المهمة هي الأسطول. يجب أن يمتلك كل task: `task_id`, `domain`, `owner surfaces`, `canonical stores`, `permissions`, `cooldowns`, `success metrics`, `frontier`, `kernel_version`. ثم يبنى رسم تضارب/تكامل بين المهام.

قواعد الملكية: مهمة Job Sprint تملك سجل الفرص المهنية، مهمة B2B تملك الحسابات التجارية، Research Leads تملك جهات البحث، Reply Monitor يملك التصنيف والردود، Artifact task يملك التصميم. لكن السجل المركزي يسمح بالاشتراك في entity مع حواف مختلفة بدل خلق نسخ متضاربة.

قاعدة عدم الازدواج: إذا وجد task آخر contact حديثاً، لا يعاد الإرسال إلا إذا تغير العرض/الدور مادياً وكان ذلك مسموحاً. **قاعدة الانتشار:** skill مثبتة مثل dedupe gate أو evidence packet تنتقل إلى كل المهام، بينما معرفة خاصة بمجال مثل CV routing لا تنتقل إلى Carousel task.

Fleet Court: كل فترة، قارن outputs: ما المهام التي تنتج دلتا صفرية؟ ما المهام التي تعيد نفس البحث؟ ما المهام التي تستهلك runtime بلا world effect؟ ما المهارات التي ظهرت في أكثر من مجال؟ ثم حدث kernels مع version/rollback.

24. هندسة المهام الكبيرة: من Mega Prompt إلى Mega Execution

المشكلة في «الميغابرومبت» أنه قد يكدس تعليمات أكثر مما يستطيع التشغيل تفعيلها. M-TAOS يقترح **Compilation over Accumulation**: بدلاً من وضع 200 قاعدة في working context دائماً، قسمها إلى دستور ثابت، kernels حسب المجال، triggers، وskills تسترجع عند الحاجة. بذلك يصبح التاريخ الطويل مصدر قدرة لا مصدر ازدحام.

المبدأ الأول: **كل تعليمات دائمة يجب أن تكون إما invariant أو trigger**. invariant مثل «لا تخمن بريدًا». trigger مثل «عند إرسال رسالة، افحص cooldown». التفاصيل الأخرى تحفظ في procedural memory وتسترجع حين يظهر نوع العقدة.

المبدأ الثاني: **لا تحوّل كل run إلى كتاب.** إذا كانت المهمة lead generation، الحجم الأمثل يظهر في عدد حسابات متحققة ورسائل جيدة وسجل نظيف، لا في شرح 20 صفحة عن lead generation. إذا كانت المهمة كتاباً، يصبح artifact نفسه world effect ويستحق حجماً كبيراً.

المبدأ الثالث: **الحوسبة الهجومية تحتاج خصماً عدائياً.** كلما زاد التوسع، خصص جزءاً متزايداً من الميزانية لاكتشاف duplicate/weak-fit/stale/unsupported. هذا هو معنى «التطوير الذاتي المتزايد الشدة»: ليست شدة الكلام، بل شدة اختبار المخرجات كلما ارتفع أثرها.

25. Deployment Modes

النمط	الاستخدام	الحوسبة	الذاكرة	الفعل
Compact	سؤال سريع	1-2 passes	working only	قليل
Research	موضوع عميق	4-8 workstreams	evidence + semantic	artifact
Opportunity	leads/jobs/collabs	broad discovery + strict verify	canonical ledger	when authorized Gmail/Drive
Production	PDF/design/code	staged build + QA	versioned artifacts	file outputs
Fleet	مهام متعددة	cross-task audit + diffusion	shared kernel/registry	prompt/config updates
PETA	مهمة معقدة ذات أدوات كثيرة	+ DAG + parallel + verifier saturation	all layers	multiple world effects

اختيار النمط ديناميكي. لا تستخدم PETA لعملية بسيطة لأن كلفة التنسيق ستتجاوز فائدته. المعيار هو قيمة المسألة وتعقيدها وقابلية التفريع.

26. نظام النسخ والتحديث

يعامل M-TAOS كل kernel كبرنامج: `MAJOR.MINOR.PATCH`. تغيير MAJOR إذا تغيرت دلالات النجاح أو الذاكرة/الأفعال، MINOR لإضافة skill/loop/tool دون كسر السابق، PATCH لإصلاح صياغة أو guardrail. لكل إصدار: changelog، سبب التغيير، المهام المتأثرة، regression cases، وخطة rollback.

تحديث knowledge لا يمحو القديم إذا كان متعارضاً، يسجل `supersedes` أو `contested_by`. تحديث suppression أو permission يملك أولوية أعلى من قواعد outreach القديمة. أما فرضية نظرية جديدة فلا تدخل kernel التنفيذي حتى تجتاز Weak Operationalization Court.

27. برنامج بحثي لتطوير علم Task OS

المشروع	السؤال	المنهج	المخرج
Adaptive Compute Curves	بناء منحنيات MCV عبر أنواع مهام: بحث، فرص، كتابة، كود.	telemetry من runs مع budget variations	سياسة تخصيص قابلة للمعايرة.
Chat-to-Kernel Benchmark	قياس فقد المعلومات عند ضغط محادثات طويلة.	corpora + reconstruction questions	تعريف KRS عملي.
Cross-Task Skill Transfer	هل skill مستخرجة من lead gen تفيد research collaboration؟	A/B task kernels	حدود الانتقال السلبي.
Branch Diversity vs Utility	هل التنوع الدلالي يساوي تنوعاً وظيفياً؟	branch embeddings + outcome	مقياس diversity مفيد.
Court Ablations	إزالة كل محكمة وقياس الأخطاء.	task suite	أي المحاكم لها أكبر أثر.
Execution-Proof Hierarchy	مقارنة نصي vs tool vs source vs functional test.	heterogeneous tasks	نظام ترجيح للدليل.
Memory Write Policies	ما الذي يجب حفظه؟	longitudinal agent runs	سياسة learned write/filter.
HyperFrontier Scheduling	مقارنة FIFO, urgency, EV, MCD routing.	recurring tasks	أعلى progress continuity.
PETA Parallelism	تحديد نقطة توازن التوازي/التنسيق.	tool-heavy workflows	parallelism boundary curves.
Outreach Outcome Optimization	تحسين من sends إلى replies/ meetings/revenue.	ethical campaign telemetry	World-Effect Yield.
Artifact Density Metric	قياس info/page دون قتل readability.	expert ratings + compression tests	cognitive density score.
AGI Task Robustness Suite	مهام بسيطة بشرياً لكنها متعددة الأدوات/ الذاكرة.	inspired by GAIA/OSWorld	benchmark خاص بـstateful recurring tasks.

28. اثنتا عشرة فرضية بحثية

1. **فرضية H1:** سياسة حوسبة تكيفية على مستوى فروع المهمة ستتفوق على budget ثابت عند مساواة التكلفة الكلية في المهام غير المتجانسة.
2. **فرضية H2:** وجود متحقق خارجي سيجعل الانعكاس الذاتي أكثر استقراراً من النقد الداخلي الصرف في المهام ذات الإجابة القابلة للفحص.
3. **فرضية H3:** ذاكرة إعادة البناء المضغوطة ستتفوق على إدخال سجل المحادثة الكامل في نسبة النجاح لكل token بعد عدد كبير من التشغيلات.
4. **فرضية H4:** تخصيص ownership/cooldowns في أسطول المهام سيخفض duplicate actions أكثر من dedupe النصي بعد التوليد.
5. **فرضية H5:** التنوع الوظيفي للوكلاء يتنبأ بتحسن الدمج أكثر من تنوع persona أو الأسلوب.
6. **فرضية H6:** Graph-based memory ستتفوق على vector-only في أسئلة «لماذا/كيف ترتبط الحالات عبر الزمن» بعد تراكم سجل كبير.
7. **فرضية H7:** قياس CAD سيعيد ترتيب ما يسمى «أفضل تشغيل» مقارنة بمقياس طول الرد أو عدد السجلات الخام.
8. **فرضية H8:** Branch-Falsify-Merge سيحسن calibration أكثر من majority vote في المجالات التي يمكن فيها ربط الفروع بأدلة مستقلة.
9. **فرضية H9:** HyperFrontier persistent scheduling سيقلل وقت إعادة الاكتشاف بين التشغيلات المتكررة بنسبة قابلة للقياس.
10. **فرضية H10:** تطبيق تحقق غير متناظر بحسب irreversibility سيحافظ على السرعة مع خفض الأخطاء الخارجية أفضل من تحقق ثابت لكل الأفعال.
11. **فرضية H11:** استخراج Failure Signatures من عدة مهام سيمنع فئات أخطاء جديدة أفضل من قواعد يدوية لكل task على حدة.
12. **فرضية H12:** التوسع إلى PETA grade سيحقق عائداً متناقصاً بعد نقطة تعتمد على جودة الموجه والمتحقق، لا على حجم النموذج وحده.

29. اثنا عشر سؤالاً مستقبلياً عميقاً

1. كيف نقدر MCV قبل صرف الحوسبة لا بعدها؟
2. هل يمكن تعلم Vascular Router من logs دون أن يتعلم مقاييس vanity؟
3. ما الحد الأدنى من الذاكرة الذي يحافظ على agency الطويلة؟
4. كيف نمثل تعارضاً زمنياً بين مصادر وحالات العالم داخل GraphRAG؟
5. متى يصبح تعدد الوكلاء أسوأ من وكيل قوي مع verifier؟
6. كيف نقيس world-effect عندما تظهر النتيجة بعد أيام أو أسابيع؟
7. كيف نفصل جودة kernel عن جودة النموذج الأساسي؟
8. ما أفضل طريقة لاختبار cross-task transfer دون تسرب معلومات؟
9. كيف نحدد expiry للذكريات والقيود؟
10. هل يمكن للوكيل أن يبني benchmarks جديدة تكشف نقاط ضعفه دون reward hacking؟
11. كيف نمثل permission كجزء من state لا كتعليمة نصية فقط؟
12. ما الذي يميز Task OS قوي عن مجرد workflow automation مع LLM؟

30. اثنا عشر مفهوماً مفتاحياً يجب ضبطه

1. **الفرق الحاكم** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
2. **الأثر العالمي** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
3. **الدليل التنفيذي** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
4. **الحوسبة الحدية** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
5. **التوازي المقيد** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
6. **ذاكرة إعادة البناء** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
7. **التعارض المحفوظ** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
8. **الـHyperFrontier** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
9. **الملكية عبر المهام** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
10. **التشبع الحوسبي** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
11. **الانتشار المهاري** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.
12. **التحقق غير المتناظر** — لا يُقبل تعريفه إلا إذا أمكن ربطه بقرار واختبار.

31. اثنا عشر خطأً شائعاً

1. اعتبار output الأكبر أفضل بالضرورة.
2. عدّ استخدام عدة agents دليلاً على multi-agent intelligence.
3. حفظ كل المحادثة واعتباره ذاكرة.

4. تشغيل أدوات كثيرة دون DAG.

5. التصحيح الذاتي بلا معلومة جديدة.

6. إرسال عدد كبير من الرسائل واعتبار ذلك conversion.

7. الاستشهاد بمصدر لا يثبت الادعاء.

8. تسمية framework جديد واعتباره نظرية.

9. توزيع compute بالتساوي.

10. نسيان قراءة نتيجة الفعل الخارجي.

11. عدم كتابة failure memory.

12. تحديث prompt من دون regression/rollback.

32. اثنا عشر تمييزاً يغير قراءة المجال

1. **الحوسبة ≠ القيمة الحدية للحوسبة** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

2. **الذاكرة الخام ≠ ذاكرة إعادة البناء** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

3. **التفريع ≠ التفريع المتنوع وظيفياً** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

4. **النقد ≠ الهدم المرتبط بـcrux** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

5. **المصدر ≠ دليل يثبت الادعاء** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

6. **الفعل ≠ أثر مع readback** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

7. **الإرسال ≠ نتيجة pipeline** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

8. **الوكيل ≠ عقدة ذات صلاحيات واختبار** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

9. **التوازي ≠ استقلال اعتمادي** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

10. **الضخامة ≠ الكثافة الإنجازية** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

11. **الاستمرار ≠ HyperFrontier** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

12. **الجدة ≠ فرق تشغيلي قابل للاختبار** — الثاني هو الصيغة التشغيلية التي يعتمدها M-TAOS.

33. النموذج الجديد: M-TAOS / M-PETA Architecture

الاسم: M-TAOS — Moussaif Task AGI-Oriented HyperOperating System.
النسخة الموسعة للتشغيل الكثيف: M-PETA — Moussaif Parallel Epistemic Tool-Action Architecture.

33.1 لماذا نحتاجه؟

الأطر الحالية تقدم قطعاً قوية: agent compilers، tool workflows، multi-agent OS، memory thoughts، search over reasoning+acting، benchmarks. لكن المهمة المجدولة طويلة العمر تحتاج طبقة تجمع **التراكم بين التشغيلات** مع **إدارة الحوسبة والكتابة الراجعة** و**تحويل النتائج إلى أثر** و**حوكمة أسطول مهام**. M-TAOS يقترح هذه الوحدة التكاملية كتركيب هندسي لا كبديل لكل إطار.

33.2 الطبقات

1. Constitution & Permission Kernel.

2. Stateful Memory Fabric.
3. Problem/Task Graph Compiler.
4. Vascular Compute Router.
5. Parallel Reasoning/Research/Tool Fabric.
6. Evidence & Destruction Courts.
7. World-Effect Action Layer.
8. Run Delta / HyperSkill Distillation.
9. Cross-Task Diffusion & Versioning.

33.3 ما الذي يفسره أفضل؟

يفسر لماذا تفشل مهمة جيدة البرومبت في الاستمرار: ليس بسبب نقص «المعرفة» فقط، بل بسبب عدم وجود سياسات لما يُسترجع، أين تُصرف الحوسبة، متى يُنفذ الفعل، ماذا يُكتب، وكيف نتعلم المهام الأخرى. ويقدم لغة لتمييز «حجم output» عن «كثافة الإنجاز».

33.4 ما الذي لا يفسره؟

لا يفسر حدود الذكاء داخل النموذج الأساسي، ولا يضمن التخطيط الصحيح في البيئات المفتوحة، ولا يحل مشكلة alignment العامة، ولا يحول scheduler إلى AGI. يحتاج إلى أدوات موثوقة، أذونات، benchmarks، وبيانات تشغيل حقيقية.

33.5 التنبؤات

إذا طبق النظام جيداً نتوقع: انخفاض التكرار، ارتفاع verified deltas/run، قصر latency في الفروع المستقلة، زيادة reuse من التاريخ، انخفاض claims unsupported، وتحسن outcomes عندما يكون world-effect قابلاً للقياس.

33.6 اختبار النموذج

شغّل نفس أسطول المهام أربعة أسابيع بbaseline ثم أربعة أسابيع بـM-TAOS، مع تسجيل tokens/tool calls/latency/records/actions/replies/errors/rework. أجر ablations للذاكرة، الموجه، المحكمة، والتوازي. لا يكفي تقييم جودة النص بواسطة نموذج آخر.

34. نسخة النموذج في صفحة واحدة

Input: Goal + Current State + Permissions + Canonical Memory.
Compile: Kernel → Problem Graph → Execution DAG.
Route: allocate compute by Value × Reducible Uncertainty × Centrality × Verifiability − Cost − Risk.
Expand: 3–10 independent workstreams when useful; diversify by function/evidence, not persona.
Act: tools are part of reasoning; read back world state.
Verify: evidence packets + execution proofs; stronger checks for irreversible actions.
Destroy: hallucination, reductionism, false novelty, weak operationalization, agentic failure, saturation.
Merge: preserve unresolved conflicts and boundary conditions.
Commit: Run Delta → canonical store → episodic/semantic/procedural/frontier memory.
Learn: extract HyperSkills/Failure Signatures; diffuse only validated invariants across tasks.
Stop: when marginal verified utility falls below threshold or runtime/tool boundary is reached.
Next: start from the HyperFrontier, not from zero.

تعريف MEGA: Multi-pass Evidence-Grounded Agency.
تعريف PETA: Parallel Epistemic Tool-Action.
المعيار الأعلى: أكبر فرق موثق في العالم لكل وحدة حوسبة، لا أكبر كمية نص.

35. محكمة الهدم الكبرى: أقوى الاعتراضات

الاعتراض 1: إعادة تسمية معمارية الوكلاء

جزء معتبر من المكونات معروف فعلاً. الجدة المدعاة محدودة إلى تركيب سياسات compute/writeback/fleet governance ومقاييسها. يجب ألا يسوق M-TAOS كاختراع لكل مكون.

الاعتراض 2: المقاييس غير معايرة

صحيح. MCV/CAD/MCD profiles هي heuristics حتى تتوفر telemetry وتجارب. الحل: إبقاؤها vector/qualitative ثم المعايرة.

الاعتراض 3: التوازي قد يزيد الأخطاء

صحيح عند وجود dependencies أو writes مشتركة. لذلك Parallelism Boundary + sync barriers شرط.

الاعتراض 4: الذاكرة قد ترسخ الهلوسة

صحيح. provenance + contradiction edges + confidence + write filter ضرورية.

الاعتراض 5: self-critique غير موثوق

صحيح بحسب أدبيات متعارضة [S26]. M-TAOS يعطي أولوية للإشارة الخارجية/المتحقق ولا يعتمد الانعكاس وحده.

الاعتراض 6: multi-agent theater

صحيح؛ تعدد personas ليس كافياً. يجب orthogonal roles وmarginal contribution.

الاعتراض 7: world-effect يرفع المخاطر

صحيح؛ يلزم permission-aware agency وasymmetric verification وsuppression/rollback.

الاعتراض 8: تكلفة التنسيق

قد تجعل M-PETA أسوأ للمهام البسيطة. لذلك توجد deployment modes وقاعدة MCV.

الاعتراض 9: ChatGPT tasks لها حدود منصة

النظام لا يزيل حدود task count، context، runtime أو أدوات المنصة. هو سياسة استغلال داخل الحدود.

الاعتراض 10: لا AGI هنا

صحيح. «AGI-oriented» يعني دعم خصائص عامة ومركبة، لا ادعاء الوصول إلى AGI.

35.1 عشر نقاط ضعف داخلية

1. تحتاج المقاييس إلى telemetry غير متاح دائماً.
2. تحديد value/uncertainty قد يعتمد على تقدير النموذج نفسه.
3. Cross-task diffusion قد يسبب negative transfer.
4. Graph maintenance يزداد تعقيداً مع الزمن.
5. الذاكرة الخارجية قد تتعارض مع privacy أو storage limits.
6. المتحقق قد يكون أضعف من المولد في مهام مفتوحة.
7. الـfrontier قد يتراكم إذا لم توجد سياسة expiry.

8. التوازي الحقيقي تحده واجهات الأدوات والمنصة.

9. بعض world effects لا يمكن قياسها فوراً.

10. التحديثات المستمرة للkernel قد تسبب drift إن غاب regression suite.

35.2 النسخة المعدلة بعد الهدم

بعد المحكمة، يجب تخفيف ادعاء «OS يزيد الذكاء» إلى صيغة قابلة للاختبار: M-TAOS **يهدف إلى رفع الأداء الوكيلي المتراكم** عبر سياسات تخصيص حوسبة، ذاكرة، تحقق، وفعل. لا يعتمد على self-critique وحده، ولا يفترض أن التوازي مفيد دائماً، ولا يختزل الأداء إلى score واحد. يصبح جوهره: **stateful bounded agency with adaptive compute and executable verification.**

36. بروتوكول إدماج M-TAOS في مهام ChatGPT

لا ينبغي لصق الكتاب كله داخل كل task prompt. المطلوب kernel مضغوط يحافظ على الفرق الحاكم. يوضع في مقدمة task الحالية دون حذف قواعد المجال. يحدد خمس التزامات:

1. **Complete-now:** لا تحول التشغيل إلى وعد أو status؛ أنجز أعلى قدر مفيد الآن.
2. **Multi-workstream:** افتح عدة فروع قيمة عندما يسمح المجال ثم نفذ المستقل منها بالتوازي المنطقي.
3. **Evidence/Action:** كل claim عالي الأثر يتحقق، وكل action يقرأ back result.
4. **Memory/Writeback:** اقرأ الحالة السابقة واكتب الدلتا في المصدر المركزي عند توفره.
5. **Court/Saturation:** اهدم، أصلح، وتوقف عند تشبع المنفعة لا عند نفاد البلاغة.

يجب الحفاظ على القواعد الأعلى الخاصة بكل task: قواعد Gmail، الخصوصية، عدم تخمين Drive، CV integrity، exact image count، emails، schema، permissions، وغيرها.

Final AGI OS Kernel 36.1

M-TAOS v1.0 — COMPACT TASK KERNEL

not as a one-shot prompt. “MEGA/PETA grade” means maximum , **Moussaif AGI-Oriented Task OS** Operate this task as a .not fake petabyte claims ,**useful verified work within the actual runtime/tool limits**

1. **Load state first:** recover relevant prior chat/task state, canonical files/ledgers, recent tool outcomes, unresolved frontier, suppression/cooldown/permissions. Do not redo resolved work.
2. **Compile the goal:** define the concrete world/artifact delta and success test; decompose into a dependency graph. Open 3–10 independent high-value workstreams when the domain supports it.
3. **Route compute adaptively:** spend more reasoning/search/tool budget on high-value, high-uncertainty, central, verifiable branches; parallelize only dependency-independent branches; stop or redirect branches whose marginal verified utility collapses.
4. **MEGA multi-pass:** discovery → verification → synthesis → falsification/quality court → action/artifact → readback → state writeback. Do not replace these passes with generic prose.
5. **PETA execution:** when useful, diversify branches by evidence/method/function, orchestrate tools, preserve disagreements, and merge through evidence rather than majority-style averaging.
6. **External signal beats empty reflection:** self-critique must seek new evidence, tool feedback, tests, counterexamples or independent branches. Avoid repetitive self-talk.
7. **World-effect loop:** when the task has standing authorization and safe available actions, execute them in the same run; verify recipient/target/state first; read the result after action. Preserve all domain-specific safety and anti-spam rules.

8. **Persistent intelligence:** record the run delta, provenance, decisions, failures, reusable assets, new HyperSkills, and next HyperFrontier in the task's canonical store when available. Update existing records rather than creating duplicates.

9. **Destruction courts:** check hallucination, stale evidence, reductionism, false novelty, weak operationalization, duplicate work, unsafe agency, and compute saturation. Repair before claiming completion.

10. **Dense chat output:** report concrete work completed, verified deltas, actions/artifacts, blockers that truly remain, and the ranked next frontier. No filler, no "background work" promises, no deliberate under-computation.

Priority law: **maximize verified world/artifact delta per unit of compute while preserving correctness, reversibility, provenance, and task-specific constraints.**

37. سجل البناء المعرفي

المصادر التي غيّرت اتجاه البناء: أدبيات test-time compute غيّرت الفكرة من «أقصى تفكير دائماً» إلى «تخصيص تكيفي وقياس المنفعة الحدية» [S06][S24][S25]. نقد التصحيح الذاتي منع الاعتماد على reflection الداخلي وحده [S26]. LLMCompiler دعم ترجمة المهمة إلى dependencies والتوازي [S18]. MemGPT/CoALA دعما فصل الذاكرة والتحكم عن prompt الخام [S08][S17]. GraphRAG/HippoRAG وسعا الذاكرة من chunks مستقلة إلى بنية علاقات [S20][S21]. Benchmarks التنفيذية مثل SWE-bench وOSWorld دفعت إلى أولوية execution proof [S15][S16].

فرضيات رُفضت أثناء البناء: أن زيادة عدد agents وحدها ترفع الجودة؛ أن self-reflection كافٍ؛ أن output الأكبر مرادف للحوسبة الأقوى؛ أن كل task يحتاج نفس مستوى التوسع؛ أن السجل الخام هو أفضل ذاكرة.

النموذج الأولي الذي حُسن: بدأ التصور كـ«موجه يطلب 3–10 فروع وعمقاً أكبر»، ثم تحول إلى نظام ذي DAG، موجه ميزانية، طبقات ذاكرة، world-effect loop، ومحاكم/telemetry، لأن التوسع وحده غير منضبط.

مناطق الجهل المتبقية: منحنيات MCV الواقعية؛ أفضل سياسة write/forget؛ قياس world-effect المتأخر؛ حدود انتقال skills؛ وتفاعل task schedulers مع قيود منصات مختلفة.

أكثر نقطة قابلة للنقد: المصطلحات MCD/M-TAOS قد تبدو إعادة تسمية ما لم تُربط بتجارب ومقاييس. لذلك يصر الكتاب على false-novelty court وبرنامج اختبار.

أقوى إضافة معرفية: دمج سؤال «كيف نزيد الحوسبة؟» مع «متى نوقفها؟» و«كيف نكتب نتيجتها بحيث تزيد ذكاء التشغيل التالي؟» داخل وحدة Task OS طويلة العمر.

38. التقييم الذاتي للجودة قبل التسليم

المعيار	التقييم / الحد	مبرر مختصر
عمق التأصيل	14/15	ربط الأدبيات ببنية تشغيلية لا مجرد سرد
جودة المصادر	13/15	مصادر قوية ومفتوحة/بحثية؛ لم تُقرأ كل الأوراق كاملة
كثافة الصفحات	10/10	تصميم نصي كثيف وجداول تشغيلية دون صفحات بطاقات كثيرة
وضوح البنية	10/10	دستور → /kernel → graph → skills/loops → tools → courts → deployment
قوة النقد	10/10	محاكم واعتراضات وحدود واضحة
الأصالة والتركيب	13/15	تركيب جديد مع تحفّظ false novelty
قابلية الاختبار	10/10	hypotheses, metrics, ablations, execution proofs
جمال PDF	5/5 مستهدف	Amiri, RTL, A4, print CSS, verified rendering

المعيار	التقييم / الحد	مبرر مختصر
عدم الحشو	5/5 مستهدف	بنية معيارية ومحتوى إجرائي
فائدة للقارئ المتقدم	5/5	kernel قابل للحقن وبرنامج بحثي
المجموع	**95/100 مستهدف**	يحتاج تجارب تشغيلية لاحقة لتحويل heuristics إلى مقاييس معيرة

39. قائمة المصادر الأساسية

ID	المؤلف	المصدر	السنة	المعرف	دوره في البناء
S01	Yao et al.	ReAct: Synergizing Reasoning and Acting in Language Models	2022/2023	arXiv:2210.03629	استدلال + فعل متداخل.
S02	Yao et al.	Tree of Thoughts: Deliberate Problem Solving with Large Language Models	2023	arXiv:2305.10601	بحث على مسارات واسترجاع/تراجع.
S03	Besta et al.	Graph of Thoughts: Solving Elaborate Problems with Large Language Models	2023	arXiv:2308.09687	تمثيل التفكير كرسم ودمج/feedback.
S04	Shinn et al.	Reflexion: Language Agents with Verbal Reinforcement Learning	2023	arXiv:2303.11366	ذاكرة انعكاسية من التغذية الراجعة.
S05	Madaan et al.	Self-Refine: Iterative Refinement with Self-Feedback	2023	arXiv:2303.17651	دورات feedback/refinement.
S06	Snell et al.	Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters	2024	arXiv:2408.03314	تخصيص compute حسب صعوبة المسألة.
S07	Schick et al.	Toolformer: Language Models Can Teach Themselves to Use Tools	2023	arXiv:2302.04761	تعلم اختيار واستعمال APIs.
S08	Packer et al.	MemGPT: Towards LLMs as Operating Systems	2023	arXiv:2310.08560	إدارة سياق افتراضي وطبقات ذاكرة.
S09	Park et al.	Generative Agents: Interactive Simulacra of Human Behavior	2023	arXiv:2304.03442	memory/reflection/planning.
S10	Wu et al.	AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation	2023	arXiv:2308.08155	بنية multi-agent مرنة.

ID	المؤلف	المصدر	السنة	المعرف	دوره في البناء
S11	Hong et al.	MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework	2023	arXiv:2308.00352	SOPs وأدوار وتدفقات عمل.
S12	Wang et al.	Voyager: An Open-Ended Embodied Agent with Large Language Models-	2023	arXiv:2305.16291	skill library + curriculum + feedback.
S13	Liu et al.	AgentBench: Evaluating LLMs as Agents	2023	arXiv:2308.03688	تقييم وكلاء عبر بيئات تفاعلية.
S14	Mialon et al.	GAIA: a benchmark for General AI Assistants	2023	arXiv:2311.12983	مهام عامة متعددة الأدوات/الوسائط.
S15	Jimenez et al.	SWE-bench: Can Language Models Resolve Real-World GitHub Issues?	2023/2024	arXiv:2310.06770	تقييم تنفيذي لمشكلات برمجية حقيقية.
S16	Xie et al.	OSWORLD: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments	2024	arXiv:2404.07972	تقييم تنفيذي عبر أنظمة وتطبيقات حقيقية.
S17	Sumers et al.	Cognitive Architectures for Language Agents	2023/2024	arXiv:2309.02427	CoALA: ذاكرة، أفعال، اتخاذ قرار.
S18	Kim et al.	An LLM Compiler for Parallel Function Calling	2023/2024	arXiv:2312.04511	تخطيط dependencies وتنفيذ أدوات متوازٍ.
S19	Hao et al.	Reasoning with Language Model is Planning with World Model	2023	arXiv:2305.14992	تخطيط/بحث بموديل عالم.
S20	Edge et al.	From Local to Global: A Graph RAG Approach to Query-Focused Summarization	2024/2025	arXiv:2404.16130	GraphRAG للأسئلة العالمية.
S21	Gutiérrez et al.	HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models	2024	arXiv:2405.14831	KG + Personalized PageRank للذاكرة.
S22	Prasad et al.	ADaPT: As-Needed Decomposition and Planning with Language Models	2023	arXiv:2311.05772	تفكيك تكيفي حسب صعوبة التنفيذ.
S23	Yang et al.	SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering	2024	arXiv:2405.15793	أهمية واجهة agent-computer.

ID	المؤلف	المصدر	السنة	المعرف	دوره في البناء
S24	Zhu et al.	Scaling Test-time Compute for LLM Agents	2025	arXiv:2506.12928	test-time scaling للوكلاء والتنويع/ verification.
S25	Hariri et al.	Test-Time Scaling in Reasoning LLMs: ,Inference Regimes Evaluation, and Reproducibility	2026	arXiv:2608.04001	تصنيف regimes ومتطلبات reproducibility.
S26	Huang et al.	Large Language Models Cannot Self-Correct Reasoning Yet	2023/2024	arXiv:2310.01798	حدود self-intrinsic correction.
S27	Liang et al.	Encouraging Divergent Thinking in Large Language Models through Multi-Agent Debate	2023	arXiv:2305.19118	التنوع والمناظرة ومشكلة Degeneration of Thought.
S28	Smit et al.	Should we be going MAD? A Look at Multi-Agent Debate Strategies for LLMs	2023	arXiv:2311.17371	نتائج مختلطة وحساسية بروتوكول المناظرة.
S29	Shen et al.	HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face	2023	arXiv:2303.17580	LLM كمنسق نماذج/ أدوات.
S30	Lewis et al.	Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks	2020	arXiv:2005.11401	الأساس الكلاسيكي لRAG.
S31	Sarthi et al.	RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval	2024	arXiv:2401.18059	استرجاع هرمي عبر شجرة summaries.
S32	Gutiérrez et al.	From RAG to Memory: Non-Parametric Continual Learning for Large Language Models	2025	arXiv:2502.14802	الانتقال من retrieval إلى memory مستمرة.
S33	Cobbe et al.	Training Verifiers to Solve Math Word Problems	2021	arXiv:2110.14168	دور verifier في اختيار حلول أفضل.
S34	Manakul et al.	SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection	2023	arXiv:2303.08896	عدم الاتساق كإشارة للهلوسة.
S35	Qin et al.	ToolLLM: Facilitating Large Language Models to Master Real-world +16000 APIs	2023	arXiv:2307.16789	توسيع tool-use على APIs كثيرة.

ID	المؤلف	المصدر	السنة	المعرف	دوره في البناء
S36	Zhou et al.	WebArena: A Realistic Web Environment for Building Autonomous Agents	2023/2024	arXiv:2307.13854	بيئة ويب واقعية للوكلاء.
S37	Bai et al.	Constitutional AI: Harmlessness from AI Feedback	2022	arXiv:2212.08073	دساتير ونقد ذاتي/تغذية راجعة للسلامة.
S38	Du	Memory for Autonomous LLM Agents: Mechanisms, Evaluation, and Emerging Frontiers	2026	arXiv:2603.07670	مسح حديث لآليات ذاكرة الوكلاء.

39.1 ملاحظة استعمال المصادر

القائمة ليست ادعاء بقراءة كاملة لكل PDF. في هذا الإصدار استُخدمت بيانات النشر والملخصات والمقاطع المتاحة عبر البحث كأساس للتأصيل المقارن. ينبغي في إصدار بحثي v2 إجراء قراءة كاملة لأهم 15–20 ورقة، استخراج الجداول/الحدود بالتفصيل، وبناء سجل claim-to-page عند الحاجة إلى نشر أكاديمي محكم.

40. الخلاصة التشغيلية

المشروع الحقيقي هنا ليس جعل ChatGPT «يكتب أكثر»، بل جعل كل تشغيل **أكثر كثافة في الفرق**. النظام القوي يتذكر ما يستحق التذكر، ينسى ما يجب نسيانه، يفتح عدة مسارات عندما توجد فائدة من التنوع، يشغّل الأدوات المستقلة دون انتظار غير ضروري، يطالب بالدليل عندما يعلو أثر الادعاء، ينفذ الفعل عندما يكون مأذوناً، يقرأ النتيجة، يهدم استنتاجه، ثم يكتب دلتا صالحة لإعادة البناء.

بهذا المعنى يصبح تاريخ المحادثة مادة تدريب سياقية لا عبئاً، وتصبح المهام المجدولة مختبرات تكرارية لا إشعارات، وتصبح الذاكرة منصة تراكم، وتصبح «الشدة» زيادة في دقة الهدم والاختبار لا ارتفاعاً في النبرة. M-TAOS لا يحل مسألة AGI، لكنه يحول أسطول المهام إلى بنية أقرب إلى ما تحتاجه وكالة عامة طويلة الأمد: حالة، تخطيط، بحث، أدوات، تحقق، فعل، ذاكرة، انتقال مهارة، وقدرة على تعديل نفسها تحت محاكم واضحة.

41. خمسون HyperRules كنظام دستوري تشغيلي

Goal Delta First 41.1

ابدأ من التغير المطلوب في العالم أو الartifact، ثم ارجع إلى النص المطلوب. إذا لم تستطع صياغة delta، فالمهمة لم تُترجم بعد إلى تشغيل.

No Re-Discovery 41.2

أي معلومة أو قرار متحقق في الذاكرة لا يعاد اكتشافه إلا إذا أصبح قديماً أو ظهرت قرينة تعارضه.

State Before Search 41.3

في المهام المتكررة، قراءة الحالة الحالية أسبق من بحث جديد واسع، وإلا سيعيد النظام إنتاج ما يملكه.

Graph Before Scale 41.4

قبل فتح عشرات الفروع، استخرج dependencies. المجم دون رسم يزيد التكرار لا الذكاء.

Parallelize Independence 41.5

التوازي حق للعقد المستقلة فقط. الكتابة إلى نفس السجل أو الاعتماد على نتيجة مشتركة تحتاج sync barrier.

Critical Path Priority 41.6

أعطِ العقد التي تحجب بقية الرسم أولوية، حتى لو كانت أقل جاذبية من فروع بحثية جانبية.

Uncertainty Must Be Reducible 41.7

لا تصرف compute على عدم يقين لا توجد طريقة واقعية لتقليله في هذا التشغيل.

Verification Scales With Consequence 41.8

كلما ارتفع أثر الخطأ أو صعوبة الرجوع، ارتفع مستوى الدليل المطلوب قبل الفعل.

External Signal Preference 41.9

إذا توفر اختبار أو مصدر أو نتيجة أداة، فهو أفضل من جولة نقد ذاتي لا تضيف معلومة.

Branch Diversity Is Functional 41.10

الفروع يجب أن تختلف في مصدر الدليل أو المنهج أو الآلية أو المخاطر، لا في الصياغة.

Merge by Evidence 41.11

لا تجمع الفروع بالمتوسط الأسلوبي؛ اربط claims بأدلتها وحدودها ثم احكم.

Preserve Productive Disagreement 41.12

إذا لم يوجد دليل فاصِل، احتفظ بالخلاف كعقدة مفتوحة مع تجربة فاصلة.

Action Needs Readback 41.13

استدعاء الأداة ليس نجاحاً. اقرأ الناتج أو حالة ما بعد الفعل وسجلها.

Canonical Store Wins 41.14

عند تعدد chat/file/sheet، عيّن مصدراً مركزياً للحالة وأنتج بقية الأسطح منه.

Stable IDs Before Volume 41.15

لا توسع dataset قبل وجود مفاتيح dedupe واستقرار للمعرفات.

Suppression Is Memory 41.16

bounce/opt-out/forbidden state ليست ملاحظة عابرة بل قيد ذاكرة عالي الأولوية.

Freshness Is Claim-Specific 41.17

ليس كل شيء يحتاج ويباً حديثاً؛ الادعاء الزمني أو المتغير يحتاجه، التعريف المستقر قد لا يحتاج.

Evidence Has Scope 41.18

مصدر موثوق يمكن أن يثبت ادعاءً ضيقاً فقط؛ لا توسع دلالته لأن سلطته عالية.

Novelty Requires Operational Delta 41.19

الفكرة الجديدة يجب أن تضيف عملية أو تنبؤاً أو تمييزاً أو أداة لا مجرد اسم.

Compression Must Regenerate 41.20

اختبر أي ملخص/ذاكرة مضغوطة بقدرتها على إعادة بناء القرار والخطة.

Memory Needs Forgetting 41.21

المعلومة القديمة أو المنسوخة أو عديمة القيمة يجب أن تنخفض أولويتها أو تنتهي صلاحيتها.

Failures Become Guardrails 41.22

الخطأ المتكرر يتحول إلى signature ثم precheck، لا إلى قصة في تقرير.

Successes Become Skills 41.23

النجاح المتكرر يتحول إلى procedure مع preconditions/tests قبل نقله لمهام أخرى.

Cross-Task Transfer Needs Scope 41.24

لا تنقل قاعدة من مجال إلى آخر قبل تعريف ما الذي يجعلها صالحة خارج مصدرها.

Budget Is Multi-Currency 41.25

احسب الزمن، استدعاءات الأدوات، الانتباه، المخاطر، السياق، لا التوكنات فقط.

Stop on Marginal Collapse 41.26

إذا لم تعد الجولة تضيف evidence/action/decision novelty، أوقفها أو انقل budget.

Minimum Pass Floor 41.27

قاعدة التشبع لا تسمح بتجاوز discovery/verification الضروريين بحجة السرعة.

High Volume Requires Higher Courts 41.28

كلما زاد عدد السجلات أو الرسائل، خصص نسبة أعلى للـdedupe والـverification.

No Quantity Fabrication 41.29

إذا لم توجد 100 نتيجة قوية، لا تصنع 100 نتيجة ضعيفة لتحقيق target عددي.

Output Must Be Typed 41.30

صنّف ما خرج: fact, lead, action, artifact, hypothesis, failure, next frontier.

Chat Is a View, Not the Database 41.31

المحادثة سطح تفاعل؛ الحالة طويلة العمر يجب أن تكون قابلة للاسترجاع والنسخ.

Tool Choice Is a Reasoning Step 41.32

اختيار الأداة يُبرر بحسب ما تستطيع إثباته أو تغييره، لا لأنها متاحة فقط.

Interface Design Matters 41.33

قدرة الوكيل تتغير بتصميم واجهة الأداة ومساحة الأفعال، لا تلوم النموذج وحده.

Replan on State Change 41.34

إذا جاءت observation تغير dependency أو الفرضية، لا تكمل الخطة القديمة آلياً.

Separate Discovery From Acceptance 41.35

المرشح المكتشف ليس lead/claim مقبولاً حتى يمر ببوابة تحقق.

Separate Activity From Outcome 41.36

عدد الأبحاث أو الرسائل نشاط؛ الاجتماعات/الردود/التغيرات المتحققة نتائج.

Record Negative Evidence 41.37

عدم وجود رد، رفض، فشل اختبار، أو مصدر ينفي الفرض معلومة يجب حفظها.

Use Edge Cases as Probes 41.38

الحالات القصوى تكشف حدود النموذج أسرع من الأمثلة المريحة.

Use Inversion to Open Domains 41.39

اقلب الافتراض الحاكم ثم اطلب ما الدليل الذي يجعله ممكناً أو مستحيلاً.

Use Cross-Domain Fusion Carefully 41.40

استعارة آلية من مجال آخر تبدأ كفرضية؛ لا تنتقل تلقائياً إلى حقيقة.

Court Must Repair 41.41

أي نقد لا ينتج إصلاحاً أو decision gate هو تكلفة بلاغية.

Version Every Kernel Change 41.42

التعديل الذي لا يمكن ربطه بإصدار/changelog يصعب تقييم أثره أو الرجوع عنه.

Regression Before Fleet Diffusion 41.43

قبل حقن rule جديدة في كل tasks، اختبر ألا تكسر قواعد خاصة بمجال حساس.

Protect Specific Constraints 41.44

الOS العام لا يلغي CV facts، safety، exact-count، أو schema الخاصة بالمهمة.

Frontier Must Be Ranked 41.45

لا تترك قائمة "next steps" غير مرتبة؛ استخدم قيمة/إلحاح/اعتماد/جاهزية.

Frontier Must Expire 41.46

الفرع القديم يعاد تقييمه أو يحذف كي لا يصبح backlog مقبرة.

Every Run Should Raise Starting Intelligence 41.47

إذا بدأ التشغيل التالي من نفس مستوى المعرفة، لم يحدث تراكم حقيقي.

AGI-Oriented Is Not AGI 41.48

استخدم المصطلح لوصف بنية عامة وقدرات تركيبية، لا كدعوى وصول إلى ذكاء عام.

Compute Claims Need Observable Work 41.49

لا تقل "أقصى حوسبة" إذا لم يظهر ذلك في فروع/أدوات/تحقق/دلتا قابلة للرصد.

The Highest Output Is Reusable Capability 41.50

أفضل نتيجة ليست النص الذي ينتهي، بل skill/tool/memory/benchmark يجعل كل نتيجة لاحقة أفضل.

42. أنماط تشغيل تطبيقية لأسطول المهام

Pattern: Job / Recruiter Sprint 42.1

الهدف ليس قائمة وظائف بل رفع احتمال مقابلات/عروض. يبدأ التشغيل بقراءة السجل والرسائل الحديثة، ثم فروع: وظائف حية، fit/CV، recruiters، replies، outreach، prior-contact، routing. الفروع البحثية تتوازى؛ الإرسال ينتظر تحقق الجهة والبريد وcooldown. Execution proof هو صفحة الوظيفة/البريد الرسمي + نجاح إرسال Gmail + كتابة الحالة. Saturation عندما تنخفض جودة الفرص الجديدة أو لا يبقى عنوان موثوق.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: B2B Revenue Engine 42.2

يقاس بالفرص التجارية المحولة، لا بعدد الشركات. الفروع: decision role، pain hypothesis، buying-power signal، account discovery، prior-contact، offer selection، personalization، official route. تُرفض الشركات التي لا تملك fit أو قناة مهنية. بعد الإرسال يصبح lead في state=contacted لا "reply/meeting/proposal/revenue". won" مراحل مستقلة.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Research Collaboration Engine 42.3

يفتح مسارات حسب المختبر/المؤسسة/برنامج البحث/مسؤول الشراكات. أهم محكمة هي confidentiality + official route + source freshness. لا ترسل نظرية غير منشورة كاملة؛ الرسالة تختبر fit وتفتح قناة. ذاكرة research يجب أن تميز automated routing عن human interest.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Reply / Inbox Monitor 42.4

هذه مهمة change detection أكثر من discovery. لا تحتاج بحثاً واسعاً إذا لم توجد إشارة جديدة. أعلى compute يذهب لتصنيف الرسالة، ربطها بالسجل الصحيح، فهم الطلب، وصياغة/إرسال رد عندما يكون غير حساس ومأذوناً. يجب منع الرد على auto-ack وتحديث next_follow_up.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Carousel / Visual Production 42.5

الحجم لا يعني 30 مفهوماً في الشريحة. M-TAOS يترجم goal إلى arc ثم ست وحدات مستقلة مع exact render contract. التخطيط والنسخ يمكن توازيهما، لكن توليد الصور يخضع لقواعد الأداة. proof هو وجود الأصول المنفصلة ونص واضح وعدم خرق الهوية. الذاكرة تحفظ motifs الناجحة وما يجب تجنبه.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Creative Music Momentum 42.6

بدلاً من تشجيع أسبوعي، يختار التشغيل dependency إنتاجية واحدة: release، visual، mix، sound design، arrangement. يفتح فروعاً تقنية/إبداعية/تعليمية ثم يدمجها في deliverable واحد قابل للتنفيذ. proof قد يكون إعدادات/structure/checklist أو ملف فعلي.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Deep Research Book 42.7

world effect هو artifact موثوق. الفروع: new applications، controversies، mechanisms، conceptual genealogy، source landscape، critique، synthesis. لا يكفي evidence packets literature summary. تغذي claim map؛ الكتاب يمر بمحكمة الهدم وrender verification.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Software Engineering Agent 42.8

ابدأ بـcompile. tests + repo state + issue إلى files/functions/tests. استخدم أدوات التنفيذ كـoracle. لا تعتبر patch مقنعاً نجاحاً حتى تمر الاختبارات أو يوجد تفسير واضح لما تعذر. failure memory تحفظ signatures مثل dependency mismatch.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Data Analysis Task 42.9

حدد السؤال وdecision use قبل الجداول. فروع: sensitivity، anomaly، model/aggregation، segmentation، definitions، data quality. execution proof هو computation reproducible. لا تسمح للLLM بتخمين أرقام غير محسوبة.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Market / News Intelligence 42.10

freshness حاكم. التوازي مفيد عبر مصادر رسمية/أخبار/بيانات، لكن الدمج يجب أن يفرق event time عن publish time. الذاكرة تحفظ baseline ثم الدلتا فقط. saturation بعد غياب تطورات جديدة ذات أثر.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Scientific Literature Radar 42.11

استخرج contradictions، seminal/latest، authors، query graph. لا تعيد كل ورقة كل run؛ frontier يركز على new papers أو unresolved mechanisms. promote paper إلى core source فقط بعد قراءة كافية.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: CRM / Archive Consolidation 42.12

world delta هو تقليل fragmentation. الفروع: provenance, conflicts، dedupe، schema mapping، source discovery. لا تحذف original archives قبل إثبات migration. proof هو canonical location + sample verification + counts.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Talent Headhunter Outreach 42.13

الكيان المركزي recruiter/mandate لا job فقط. تحقق من current role/company/public route. استخدم رسالة قصيرة مرتبطة بتخصص واضح، سجل CV variant، ثم follow-up policy. لا ترسل عدة CVs بلا طلب.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Long-Form Content Factory 42.14

ابدأ research kernel ثم parallel branches. content graph لإحصاءات، أمثلة، اعتراضات، hooks؛ لكن final voice/claim governance مركزي. outputs الاجتماعية Views من source KB لا مصادر حقيقة مستقلة.

مخرج run القياسي: Verified Delta + Actions/Artifacts + State Writeback + Failure/Skill update + Ranked HyperFrontier.

Pattern: Knowledge Base Evolution 42.15

كل run يختار gap أو conflict بدلاً من إضافة صفحات عشوائية. new chunks تحتاج metadata/relations. court يختبر whether KB became more retrievable/operable, لا أطول.

مخرج run القياسي: Ranked HyperFrontier + State Writeback + Actions/Artifacts + Failure/Skill update + Verified Delta.

Pattern: General Opportunity Capture 42.16

أي فرصة جديدة تمر عبر state → action → expected value → contactability → authority → relevance → entity resolution. هذا يسمح بجمع jobs/collabs/grants/clients ضمن ledger واحد مع record_type مختلف دون خلط semantics.

مخرج run القياسي: Ranked HyperFrontier + State Writeback + Actions/Artifacts + Failure/Skill update + Verified Delta.

43. Benchmark Suite لنظام Task OS

ID	الاختبار	معيار النجاح
B01 Kernel Reconstruction	أعطِ وكيلًا جديداً kernel المضغوط فقط واطلب متابعة task. قِس القرارات الصحيحة مقارنة بالسجل الكامل.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B02 Re-Discovery Cost	أخفِ جزءاً من الذاكرة وقِس كم بحثاً مكرراً يحتاجه الوكيل لاستعادة ما كان معلوماً.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B03 Dependency Accuracy	قارن DAG المتوقع بتسلسل dependencies الفعلي الذي كشفه التنفيذ.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B04 Parallel Safety	شغّل عقداً مستقلة ومتداخلة؛ قِس latency والتعارضات/duplicate writes.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B05 Compute Allocation	قارن fixed budget مع adaptive routing عند نفس التكلفة.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B06 Saturation	أضف passes بعد نقطة الاستقرار وقِس novelty/evidence/action per pass.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B07 Evidence Entailment	خذ claims عالية الأثر واطلب من reviewer مستقل تقييم هل المصدر يثبتها.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B08 Freshness	اخلط مصادر قديمة وحديثة في سؤال زمني وقِس اختيار الدليل الصحيح.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B09 Hallucination Court	أدخل ادعاءات مزروعة بلا مصدر وانظر هل المحكمة تلتقطها.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B10 False Novelty	قدّم أسماء جديدة لمفاهيم قديمة وقِس قدرة النظام على اكتشاف prior-art equivalence.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B11 Operationalization	قدّم نظرية جميلة بلا input/output/test واطلب تحويلها أو رفضها.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B12 Tool Routing	وفر عدة أدوات متداخلة القدرات وقِس الاختيار بحسب evidence/cost/reversibility.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B13 Action Readback	اجعل tool call ينجح تقنياً لكن الأثر النهائي يفشل؛ هل يكتشف النظام الفرق؟	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B14 Permission Boundary	اختبر أفعالاً واضحة الإذن وأخرى حساسة؛ قِس over/under-agency.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B15 Dedupe Ledger	أدخل نفس entity عبر صيغ أسماء وروابط مختلفة وقِس duplicate prevention.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.

ID	الاختبار	معيار النجاح
B16 Suppression Memory	بعد opt-out/bounce، أعد ظهور entity في discovery وقِس منع الإرسال.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B17 Cross-Task Conflict	اجعل مهمتين تكتشفان نفس contact مع عروض مختلفة وقِس ownership arbitration.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B18 Skill Transfer	استخرج HyperSkill في domain A ثم اختبرها في B وC دون تفاصيل المصدر.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B19 Failure Recurrence	كرر trigger خطأ معروف بعد تسجيل failure signature وقِس الوقاية.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B20 Frontier Continuity	قارن تشغيلات تبدأ من ranked frontier مقابل تشغيلات تبدأ من prompt فقط.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B21 Longitudinal Memory	بعد 20 run، اسأل عن /chronology/decisions changes وقِس الدقة.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B22 Graph Retrieval	قارن vector-only وgraph-assisted في سؤال متعدد العلاقات.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B23 Artifact Render	اختبر PDF/slide/document بrender وليس وجود الملف فقط.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B24 World-Effect Yield	تتبع actions حتى outcomes المتأخرة، لا immediate send/write فقط.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B25 MEGA Grade	تحقق أن run يعبر /discovery/verification synthesis/court/action/writeback.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B26 PETA Grade	تحقق من parallel branches, dependency graph, tool orchestration, merge, cross-task memory.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B27 Noise Penalty	زد candidate count مع الحفاظ/خفض الجودة؛ هل MCD/CAD يعاقبان الضجيج؟	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B28 Countermodel Robustness	أدخل تفسيراً منافساً أقوى من الأول؛ هل يحتفظ النظام به حتى دليل فاصِل؟	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B29 Version Regression	حقن kernel جديد ثم replay حالات حرجة؛ قِس فقد القواعد الخاصة.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.
B30 Human Attention Efficiency	قِس كم سؤال/تأكيد بشري طُلب مقابل ما كان ضرورياً فعلاً.	نجاح وظيفي قابل للرصد مع baseline/ablation، لا تقييم بلاغي فقط.

44. ثلاثون Failure Signatures وذاكرة الوقاية

التوقيع	العَرَض	ذاكرة الوقاية
F01 Repeated Search	نفس query/entity يعود في runs متتالية	stable ID + prior-search timestamp + freshness trigger
F02 Duplicate Send	recipient contacted inside cooldown	Gmail/ledger precheck + suppression
F03 Stale Lead	role/opportunity expired	evidence_date + live-source validation
F04 Guessed Contact	email pattern غير رسمي	official/public-source requirement
F05 Claim Stretch	المصدر أضيق من الادعاء	claim-source entailment gate
F06 Reflection Drift	النقد يغير جواباً صحيحاً بلا evidence	external-signal rule
F07 Branch Clone	الفروع paraphrases	diversity meter + orthogonal role prompt

التوقيع	العَرَض	ذاكرة الوقاية
F08 Premature Merge	اختلافات مهمة تم مسحها	conflict graph + discriminator
F09 Tool Cascade	استدعاءات متسلسلة غير ضرورية	DAG/parallel compiler
F10 Parallel Collision	فروع تكتب نفس state	write ownership + sync barrier
F11 Context Flood	السياق ممتلئ بتاريخ غير مفيد	kernel + retrieval policy
F12 Memory Poison	معلومة خاطئة تمت ترقيتها	confidence/provenance/conflict edges
F13 Forgotten Constraint	قاعدة حاكمة ضاعت في الضغط	kernel invariant tests
F14 No Readback	action counted without outcome	mandatory post-action observation
F15 Vanity Metric	success=count instead of outcome	World-Effect Yield
F16 Artifact Exists But Broken	ملف أنشئ بلا render/test	artifact QA loop
F17 Source Monoculture	كل الفروع من نفس المصدر	source diversity floor
F18 Over-Parallelization	تنسيق أغلى من التسلسل	parallelism boundary + MCV
F19 Under-Compute	run ينتهي بعد ملخص صغير	minimum MEGA passes for complex task
F20 Compute Theater	كلام عن العمق دون workstreams	observable compute ledger
F21 Frontier Cemetery	next queue تتراكم دون إغلاق	expiry + reprioritization
F22 Version Drift	task prompts diverge silently	kernel version registry
F23 Negative Transfer	rule مشتركة تضر domain	scope tags + rollback
F24 Over-Verification	actions harmless تتعطل بتأكيدات غير لازمة	asymmetric verification
F25 Under-Verification	فعل حساس يعتمد على مصدر واحد	risk-tier verification
F26 Multi-Agent Theater	أدوار مختلفة لفظياً فقط	marginal contribution check
F27 False Theory	renaming presented as science	prior-art + predictions + experiments
F28 Unbounded Book	page target creates filler	density/saturation court
F29 Hidden Unknown	عدم اليقين يختفي في final merge	epistemic tags + unresolved nodes
F30 No Skill Extraction	نجاحات تتكرر يدوياً	procedural distillation after repeated pattern

45. مصفوفة Compute × Evidence × Agency

يمكن فهم الأنظمة الوكيلية عبر ثلاثة محاور مستقلة نسبياً: **Compute Depth**، **Evidence Strength**، **Agency Level**. الخطأ هو رفع محور واحد واعتباره ذكاءً عاماً.

الحالة	Compute	Evidence	Agency	الخطر	سياسة M-TAOS
C1	منخفض	منخفض	منخفض	جواب سطحي	توسع discovery
C2	عال	منخفض	منخفض	هلوسة مصقولة	routing /evidence court
C3	منخفض	عال	منخفض	بحث جيد بلا تركيب	synthesis/graph
C4	عال	عال	منخفض	تقرير قوي بلا أثر	action/artifact compile
C5	منخفض	منخفض	عال	فعل خطر	block/verify

الحالة	Compute	Evidence	Agency	الخطر	سياسة M-TAOS
C6	عال	منخفض	عال	أخطر حالة: agency + بلا overthinking ground	severe verification gate
C7	متوسط	عال	عال	تشغيل فعّال	readback/writeback
C8	تكيفي	متعدد ومستقل	مأذون ومقاس	الحالة المستهدفة	MEGA/PETA + saturation

الغاية ليست دائماً C8؛ المهمة البسيطة قد تحل بكفاءة في C3 أو C4. الذكاء التشغيلي هو اختيار المستوى المناسب.

46. Knowledge Cores كنواة قابلة للتوسع

Core 1 — Goal Semantics

تمثيل الغاية والتغير ومعايير النجاح والفشل. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 2 — State Semantics

تمثيل ما هو صحيح الآن، ما تغير، وما هو غير معروف. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 3 — Evidence Semantics

نوع المصدر، نطاق الإثبات، الثقة، الزمن، التعارض. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 4 — Task Graph Semantics

العقد، الاعتمادات، التوازي، الحجب، critical path. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 5 — Compute Economics

الميزانية متعددة العملات، allocation، saturation، MCV. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 6 — Memory Architecture

working/episodic/semantic/procedural/frontier. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 7 — Tool Semantics

قدرات الأدوات، evidence strength، reversibility، permissions، schemas. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 8 — Agency Governance

متى ينفذ، متى يسأل، متى يمنع، كيف يقرأ النتيجة. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 9 — Verification

source checks, functional tests, independent branches, courts. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 10 — Skill Learning

استخراج procedures من runs واختبار انتقالها. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 11 — Fleet Governance

ownership, dedupe, diffusion, task registry, canonical stores. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 12 — Versioning

kernel evolution, regression, rollback, migration. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 13 — Opportunity Systems

entity→fit→contact→action→outcome pipelines. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 14 — Artifact Systems

author→render→inspect→repair→deliver. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 15 — Research Opening

negative space, countermodels, hypotheses, experiments. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 16 — Epistemic Safety

ranks, provenance, uncertainty, source honesty. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 17 — Destruction

hallucination/reduction/novelty/operation/agency/saturation courts. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 18 — Evaluation

benchmarks, telemetry, execution proofs, ablations. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

Core 19 — Cross-Task Intelligence

shared invariants and scoped transfer. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

persistent queue and continuation intelligence. هذا الـCore يجب أن يمتلك tests، retrieval triggers، schema، وversion مستقل داخل أي implementation فعلي.

47. بروتوكول PETA لحدود التشغيل الواقعية

عندما يطلب المستخدم «ادفع الحوسبة إلى حدودها»، لا توجد رخصة لتجاهل حدود المنصة. PETA يتعامل مع الحد كمسألة scheduling: إذا كان runtime قصيراً، يُقدّم critical path والفروع ذات أعلى EV ثم يُحفظ frontier. إذا كان عدد الأدوات محدوداً، تجمع الاستدعاءات المتشابهة في batches. إذا كان context محدوداً، يجري kernelization وretrieval. إذا كانت المهام النشطة محدودة العدد، يُدمج controller logic داخل prompts الموجودة بدلاً من الادعاء بإنشاء controller مستقل غير ممكن.

هكذا تصبح الحدود جزءاً من الذكاء: النظام الذي يعرف أن لديه خمس فتحات مهام نشطة يتخذ قراراً مختلفاً عن نظام يتصرف كأن scheduler غير محدود. والضغط إلى الحد يعني الوصول إلى **runtime frontier** الفعلي بأكبر verified delta، ثم commit واضح، لا وعداً بأن «العمل سيستمر في الخلفية».

48. دستور كتابة المخرجات الضخمة

1. كل فصل كبير يبدأ بسؤال أو فرق لا بمقدمة خطابية.
2. الجداول تستخدم عند وجود أبعاد مقارنة؛ لا تحول كل شيء إلى table لتكثيف بصري وهمي.
3. كل 2–4 صفحات يجب أن تنتج أداة أو قاعدة أو تمييزاً أو اختباراً أو خريطة.
4. لا تعيد تعريف المفهوم إلا إذا تغير مستوى التجريد.
5. الأمثلة ليست زخرفة؛ يجب أن تختبر boundary أو توضح mechanism.
6. كل نظرية جديدة تتضمن test + failure + operational delta + prior-art relation.
7. كل claim حديث يحتاج مصدر حديث عند كونه زمنياً.
8. كل artifact يُرندر ويفحص.
9. لا تطارد page target بعد تشبع المعنى؛ أضف appendices عملية فقط.
10. المخرج النهائي يجب أن يحتوي compact kernel يسمح بتحويل الكتاب من معرفة إلى تشغيل.

49. ملخص تنفيذي من صفحتين — النسخة المكثفة

الصفحة التنفيذية الأولى: لماذا Task OS؟

أكبر فجوة في استعمال النماذج اللغوية للمهام المتكررة ليست نقص القدرة على كتابة جواب جيد، بل انقطاع الاستمرارية بين التشغيلات. النموذج قد يبحث اليوم ويعيد البحث غداً، يرسل رسالة ولا يقرأ أثرها، يكتشف قاعدة ولا يحولها إلى skill؛ يملك تاريخ chat ضخماً لكنه لا يعرف ما الذي يجب استرجاعه. M-TAOS يعيد تعريف المهمة كعملية ذات حالة، ويجعل الذاكرة والحوسبة والأدوات والتحقق أجزاء من نظام واحد.

القاعدة المحورية: **أقصى حوسبة نافعة ≠ أقصى طول استدلال.** نحتاج موجهًا يخصص compute للفروع التي يمكن أن تغير القرار. Test-time scaling يوفر أساساً لفكرة أن inference budget يمكن أن يغير الأداء، لكنه يبين كذلك أن السياسات تختلف باختلاف صعوبة المسألة والبروتوكول. لذلك يضيف M-TAOS مقياس المنفعة الحدية للحوسبة، ويوقف أو يحول الفروع عندما لا تنتج evidence/action novelty.

المهمة تُترجم إلى DAG. العقد المستقلة تُنفذ بالتوازي، والعقد الحرجة تأخذ أولوية. الفروع لا تُنشأ لمجرد التنوع، بل لتغطية مصادر أو آليات أو مخاطر مختلفة. بعد التوليد لا يأتي "ملخص" بل محكمة: هل الدليل يثبت؟ هل يوجد countermodel؟ هل هناك duplicate؟ هل الفعل مأذون؟ هل التوازي آمن؟ ثم world-effect: رسالة، ملف، تحديث، اختبار، أو قرار. وأخيراً writeback يكتب الدلتا في canonical state.

الصفحة التنفيذية الثانية: كيف يصبح الأسطول أذكى؟

عند تكرار النظام، تبدأ الفائدة المركبة. Memory Agent يحول runs إلى episodic memory؛ Semantic Agent يدمج ما استقر؛ Skill Distiller يحول procedures المتكررة إلى HyperSkills؛ Failure Memory يحول الأخطاء إلى guardrails؛ HyperFrontier يمنع restart؛ Cross-Task Diffusion ينقل invariants المختبرة إلى tasks أخرى مع scope وrollback.

تُقاس الجودة عبر Cross-Task Transfer، Duplicate Avoidance، World-Effect Yield، CAD، Evidence Governance، KRS وغيرها. لا تُختزل كلها في رقم واحد. ويجب قياس task OS بbenchmarks تنفيذية وطويلة المدى، على طريقة الروح التي تدفع GAIA/SWE-bench/OSWorld: هل أنجز النظام المهمة فعلاً؟

الخلاصة: M-TAOS ليس وعداً بـAGI، بل هندسة تجعل المهام أكثر شبهاً بنظام معرفي-وكلي: يعرف هدفه، حالته، ذاكرته، ميزانيته، أدواته، حدود أذوناته، معيار نجاحه، وما الذي يجب أن يتعلمه بعد كل تشغيل. رتبة MEGA تعني multi-pass evidence-grounded agency؛ رتبة PETA تضيف parallel epistemic tool-action orchestration. كلاهما يرفض التضخم غير المتحقق.

Final AGI OS Kernel — النسخة القابلة للحقن

M-TAOS v1.0 — COMPACT TASK KERNEL

not as a one-shot prompt. “MEGA/PETA grade” means maximum , **Moussaif AGI-Oriented Task OS** Operate this task as a .not fake petabyte claims ,**useful verified work within the actual runtime/tool limits**

1. **Load state first:** recover relevant prior chat/task state, canonical files/ledgers, recent tool outcomes, unresolved frontier, suppression/cooldown/permissions. Do not redo resolved work.
2. **Compile the goal:** define the concrete world/artifact delta and success test; decompose into a dependency graph. Open 3–10 independent high-value workstreams when the domain supports it.
3. **Route compute adaptively:** spend more reasoning/search/tool budget on high-value, high-uncertainty, central, verifiable branches; parallelize only dependency-independent branches; stop or redirect branches whose marginal verified utility collapses.
4. **MEGA multi-pass:** → discovery → verification → synthesis → falsification/quality court → action/artifact → readback state writeback. Do not replace these passes with generic prose.
5. **PETA execution:** when useful, diversify branches by evidence/method/function, orchestrate tools, preserve disagreements, and merge through evidence rather than majority-style averaging.
6. **External signal beats empty reflection:** self-critique must seek new evidence, tool feedback, tests, counterexamples or independent branches. Avoid repetitive self-talk.
7. **World-effect loop:** when the task has standing authorization and safe available actions, execute them in the same run; verify recipient/target/state first; read the result after action. Preserve all domain-specific safety and anti-spam rules.
8. **Persistent intelligence:** record the run delta, provenance, decisions, failures, reusable assets, new HyperSkills, and next HyperFrontier in the task’s canonical store when available. Update existing records rather than creating duplicates.
9. **Destruction courts:** check hallucination, stale evidence, reductionism, false novelty, weak operationalization, duplicate work, unsafe agency, and compute saturation. Repair before claiming completion.
10. **Dense chat output:** report concrete work completed, verified deltas, actions/artifacts, blockers that truly remain, and the ranked next frontier. No filler, no “background work” promises, no deliberate under-computation.

Priority law: **maximize verified world/artifact delta per unit of compute while preserving correctness, reversibility, provenance, and task-specific constraints.**