CAICT 中国信通院

集智 专题报告

驾驭工程：赋能智能原生软件工程研究报告

（2026 年）

中国信息通信研究院云计算与数字化研究所

2026年9月

版权声明

本报告版权属于中国信息通信研究院，并受法律保护。转载、摘编或利用其他方式使用本报告文字或者观点的，应注明“来源：中国信息通信研究院”。违反上述声明者，本院将追究其相关法律责任。

前　言

当前，人工智能正以前所未有的速度重塑软件工程的技术范式与产业格局。以大语言模型和智能体为代表的新一代智能化技术已从实验室的概念验证走向生产环境的深度应用，推动软件研发生命周期从“人为主导”向“人机协同”加速演进。软件工程正处于从“工具集成期”向“流程自治期”过渡的关键阶段，技术落地的瓶颈已从算法能力的提升，转向工程化闭环的构建、数据飞轮的运转以及组织架构的适配，业界迫切需要一套能够承接 AI 能力、规范智能行为、保障工程可靠性的新型工程范式，以支撑智能原生软件从技术原型走向规模化生产落地。

中国信息通信研究院长期跟踪人工智能与软件工程前沿动态与产业实践。本报告聚焦智能原生软件工程的核心痛点，系统性拆解驾驭工程（Harness Engineering）的概念内涵、三代技术演进路径与价值定位，全面分析全球技术发展格局、产业生态现状与核心挑战，提出驾驭工程的六大设计原则与核心技术体系，构建 L1-L5 能力成熟度标准化分级模型，深入阐述驾驭工程对软件研发全生命周期的重塑作用、对质量安全治理体系的赋能价值，以及在通用行业与重点垂直领域的规模化落地路径，最后对未来发展趋势进行展望，并提出针对性发展建议。

驾驭工程不是传统开发工具的线性迭代，也不是大模型能力的简单辅助，而是智能原生时代软件工程的核心管控框架与生态底座。它通过构建智能体运行的约束规则、反馈闭环与治理体系，实现了从“人

指挥机器”到“机器在边界内自主执行”的范式转变。立足我国超大规模市场优势、完备产业体系与丰富应用场景，加快驾驭工程技术创新与产业落地，将有效破解智能软件规模化落地的工程化难题。同时，驾驭工程作为新兴工程范式仍处于快速演进之中，本报告所述内容仅代表当前阶段的相关判断，将在持续的技术演进与产业实践中检验、迭代与完善。中国信息通信研究院愿与产业界、学术界携手共进，共同推动驾驭工程体系化建设与标准化发展，为我国加快实现高水平科技自立自强贡献力量。

目 录

图 目 录

表 目 录

CAICT 中国信通院

一、背景与总体概述

（一）人工智能重塑软件工程范式，驾驭工程应运而生

1.政策与市场双重驱动智能化研发进入深水区

全球范围内，人工智能驱动的生产力变革已成为各国共识，软件工程作为数字经济的关键基础支撑领域，其智能化转型被主要经济体普遍列为重点战略方向。美国依托《人工智能行动计划》布局开源生态、全栈 AI 工具与研发基础设施，争夺智能体工程标准主导权，欧盟借助《人工智能法案》推行分级风险管控，要求高风险 AI 开展合规审计，英国、日本、韩国相继完善配套治理规则，形成全球政策竞逐格局。当前，我国软件工程智能化转型已在政策引导与市场需求的协同作用下，从技术探索阶段进入规模化落地与深度应用的关键时期。

国家战略布局持续深化，为软件工程智能化发展构建了清晰的制度框架与实施路径。2025 年 8 月国务院印发《关于深入实施“人工智能+”行动的意见》明确将“智能研发”列为重点突破领域，提出发展面向软件全生命周期的智能化工具，构建人机协同新型研发模式。一方面，部委从产业与科研两端配套发力，2026 年 4 月工信部部署“人工智能+软件”专项行动，推进智能编程落地，培育模型即服务、智能体即服务新业态，科技部同步设立人工智能专项课题，打通技术攻关与产业转化链路。另一方面，《北京市加快建设信息软件产业创新发展高地行动方案》《四川省加快推进人工智能场景应用工作方案》等地方性配套行动计划相继出台，形成纵向衔接推动顶层设计加速向

区域实践落地转化。自上而下形成“国家顶层统筹、部委专项推进、地方场景承接”的多层次政策体系，核心导向是加速智能原生软件工程技术从试点走向规模化产业落地。

产业实践层面，企业对研发智能化的价值认知已从概念验证转向实效验证。中国信息通信研究院调研数据显示，截至 2026 年 5 月，已有 67.3%的软件相关企业将 AI 能力投入生产环节，31.2%的主体完成智能体与核心研发流程的内嵌部署。腾讯超 5 万研发人员规模化使用 AI 编程工具，AI 生成代码占比超 35%，通义灵码在车载系统开发中实现软件开发周期缩短 25%。中兴自研研发智能体落地后，代码评审时长减少 50%，AI 代码采纳率稳定至 30%。放眼中长期，IDC 预测到 2027 年智能体自动化将增强 40%以上的企业应用能力，重塑三分之一的业务流程。多方数据相互印证，AI 正从边缘试探走向业务主干，软件工程智能化已进入规模化渗透与流程级赋能的实质阶段。

2.软件研发生命周期向人机协同生态演进

软件工程的发展史本质是一部持续应对系统规模与复杂性挑战、迭代升级工程范式的变革史，此前行业已先后完成三轮系统性范式跃迁，每一次变革均回应了特定发展阶段的核心矛盾，推动软件产业实现量级式增长：**结构化与面向对象阶段**（1968 年—2000 年），以瀑布模型为核心，通过标准化流程与模块化设计化解了“软件危机”，将软件开发从个体化手工劳作模式升级为系统化工程学科。**敏捷开发阶段**（2001 年—2010 年），以迭代模式适配快速多变的业务需求。**开发运维一体化（DevOps）与云原生阶段**（2011 年—2022 年），将

解决重心从需求侧转向供给侧，打通开发与运维组织壁垒，依托容器化与自动化流水线，实现基础设施的标准化治理与软件的高频次交付。

2023 年起，大语言模型与 AI 智能体技术的爆发式突破，首次赋予 AI 端到端参与需求分析、代码生成、测试验证、运维部署全流程的工程能力，使“AI 驱动开发”从技术概念落地为可规模化复制的生产实践。这一变革打破了前三代范式“以人类工程师为唯一核心”的底层逻辑，从根本上重构了软件开发的主体关系、决策模式与治理体系，将软件工程推向智能原生范式变革的历史临界点。在智能原生时代，大模型正逐步接管代码的具体实现，人类核心工作从“手工编写代码”向“高阶编排意图”转变，这一逻辑重塑不仅指数级提升了代码生成效率，更为复杂系统的快速迭代开辟了全新路径。

3.智能软件工程范式下驾驭工程的必然性

智能化研发纵深推进叠加全域产业约束升级，从技术演进、产业落地、合规治理三重维度共同催生驾驭工程相关理论与落地体系，成为智能原生软件工程迭代演进的客观必然。

技术迭代倒逼工程体系升级，驾驭工程成为智能原生软件工程的“治理层”。传统依赖人工流程管控的工程框架，在流程规范、资源调度与风险约束等维度难以适配 AI 自主参与研发的新特征，亟需新型工程方法论承接技术变革诉求。而驾驭工程围绕智能体全生命周期，构建运行环境、约束准则与反馈闭环，跳出 AI 工具辅助的浅层模式，在人类划定的管控边界内使智能主体可靠承接复杂任务，为智能原生软件的产业化落地构筑工程基础。

产业规模化落地存在刚性缺口，驾驭工程成为智能原生软件从技术原型走向工程化生产的“转化器”。现阶段国内智能软件开发普遍面临原型落地难、版本迭代无序、全链路风险难以量化管控等共性痛点，多数大模型开发成果停留在实验室验证阶段，本质是缺少覆盖全研发链条的工程支撑，驾驭工程依托全流程管控框架，能够针对性补齐产业工程化落地短板，打通技术原型向商用产品转化的关键链路。

行业合规与高质量发展提出新要求，驾驭工程与智能原生软件工程共同构成了“管控框架－软件形态－工程范式”的三位一体。智能软件广泛渗透金融、制造、政务、医疗等关键领域，安全合规、可信可控要求持续收紧，传统软件工程的分段式管控模式难以覆盖智能代码生成、智能体自主调度等新增风险环节。以驾驭工程为体，提供约束与治理的骨架，智能原生软件为用，承载 AI 能力的价值释放，智能原生软件工程为法，定义整体的方法论与实施路径。三者相互依存、彼此支撑，共同构筑起智能原生时代的完整工程闭环。

（二）驾驭工程核心定义及三代技术演进路径

1.驾驭工程的概念内涵

驾驭工程（Harness Engineering）中的“Harness”本意为“马具”，即套在马身上用以控制引导的整套器具。Anthropic 工程师将其引入智能体领域，提出范式转变，不再在开发阶段用规则逐条指挥 AI，而是在运行时让模型充分发挥推理能力，仅提供必要的支撑与约束逻辑。**作为一种正式的工程范式概念，驾驭工程是指一种为智能体构建完整运行环境、约束规则与反馈闭环的系统工程方法，使 AI 能够在人类**

设定的边界内自主、可靠、可持续地完成复杂任务。以此为方法论，所构建出的智能原生软件其核心是将 AI 深度嵌入软件开发生命周期（SDLC）的全流程环节，使 AI 能够自主或半自主地完成全生命周期中绝大部分核心任务，实现从“程序调用智能”到“智能驱动执行”的逆向架构变革。

驾驭工程之所以构成一个独立的工程范式，在于其实现了三项根本性的范式转变：其一，智能体从“被动的代码生成器”转变为“主动的工程参与者”，这要求工程体系必须具备面向非确定性主体的治理能力，而传统 DevOps 的治理对象始终是确定性程序。其二，工程管控的重心从“流程节点”下沉至“运行时行为”，智能体编排、沙箱隔离、审计追溯等能力在驾驭工程的框架下不再彼此独立，而是围绕“约束定义→隔离执行→反馈闭环”主线一体化集成。其三，区别于传统“以流程为骨架”的静态治理，治理逻辑由流程驱动转为约束边界+反馈驱动。正是这三大范式转变，使驾驭工程成为智能原生时代不可回避的工程刚需。

来源：中国信息通信研究院

图 1 从开发时编排到运行时驾驭的范式转变

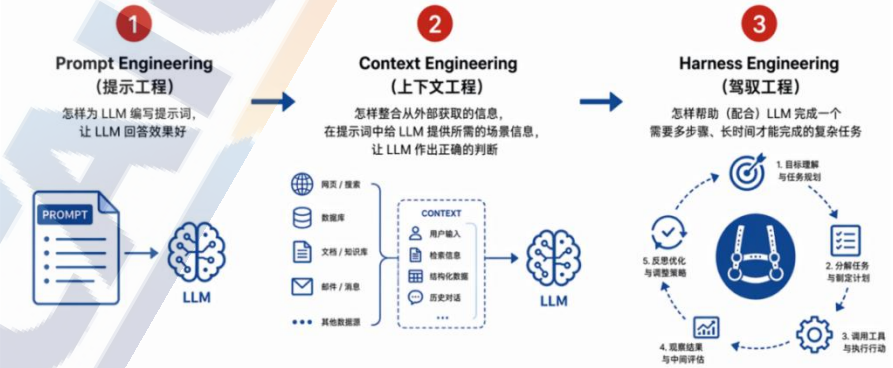
2.从提示工程到驾驭工程的层级跃迁

大模型赋能软件工程的实践，经历了从提示工程到上下文工程，再到驾驭工程的清晰演进过程，三者之间存在清晰的演进脉络：

提示词工程（Prompt Engineering），聚焦于输入侧优化，通过结构化地组织提示词引导模型，以提升大模型响应的质量。典型实践包括少样本提示、思维链引导、角色设定等，具有明显的单轮、无状态和强经验依赖属性，难以支撑多步骤、跨系统的复杂任务。

上下文工程（Context Engineering），将关注点从“单次输入”扩展到“模型所见的信息流”。通过整合历史交互、场景参数与外部检索信息构建更完整的决策语境，代表性技术包括检索增强生成、上下文窗口管理、记忆机制设计等。但面对长链条、跨系统智能体任务，其缺少行为约束与反馈闭环，无法保障执行可靠。

驾驭工程核心从“优化模型输入”转向“构建外部管控体系”。三大核心区别：目标由优化单点输出转为保障端到端任务落地、模式由开放生成变为受控执行、运行方式由人工辅助升级为边界内智能体自主协同，适配复杂长周期任务。



来源：中国信息通信研究院

图 2 从提示工程到驾驭工程的演进

（三）价值定位：智能原生软件工程的核心管控框架与生态底座

在智能原生时代，驾驭工程作为连接 AI 模型能力与工程化落地的核心桥梁，其价值定位已超越传统软件工程工具与方法论，成为适配软件全生命周期、全要素、全场景的核心管控框架与产业生态底座，其核心承载“统筹协同、规范治理、效率提升、安全保障”四大职能。

统筹协同：基于语义互操作的跨系统编排，系统级中枢的核心职能。驾驭工程以“人机协同、全生命周期驾驭”为核心逻辑，统筹协同正是这一逻辑的首要落地，围绕人机协同、全生命周期驾驭思路，搭建统一工程语义层，制定模型通信契约，依托事件驱动架构打通异构模型、工具与数据，通过全局状态视图统一调度，整合分散 AI 能力形成完整工程流水线，破除多主体集成壁垒，消除资源冲突与上下文割裂，实现 AI 与工程深度融合。

规范治理：面向全生命周期的可审计策略框架，也是其区别于传统工程范式的关键。规范治理负责对智能模型行为、数据流转实施标准化管控，涵盖模型准入与输出规范、版本控制与追溯机制、合规校验引擎三方面。在释放智能体自主决策空间的同时划定可解释、可追溯行为边界，保障智能软件标准化、安全化落地。

效率提升：从模型迭代到工程行动的时间压缩，让 AI 能力高效转化为工程实效。区别于传统流程中模型输出需人工转换，驾驭工程通过构建 AI 导向的 MLOps 扩展流水线、优化推理通道与批处

理、实施增量学习，并结合智能资源调度算法动态分配计算资源，实现从模型迭代到工程行动的时间压缩与单位计算资源的最大产出。

安全保障：对抗性与容错性的内生设计，实现“安全驾驭”的核心保障。驾驭工程将安全嵌入决策链路的原生机制，内置对抗鲁棒性检测识别异常输入，配置安全围栏拦截风险行为，配套容错降级机制保障核心业务连续运行。全链路安全事件留痕并可视化展示，支持事后审计归因，贯穿软件全生命周期。

来源：中国信息通信研究院

图 3 驾驭工程核心管控框架与生态底座示意图

这一框架化定位的独特价值体现在两个层面。其一，驾驭工程的本质是治理框架而非工具产品。将约束 AI 行为的成熟模式沉淀为标准化的治理组件与互操作契约，使不同组织的智能体在统一治理语义下协同运转。治理框架提供的约束逻辑与接口规范具备跨越技

术周期的稳定性，构成 AI 工程化长期发展的底层支撑。**其二，作为产业级管控框架，驾驭工程承担着行业治理标准互认的枢纽角色。**通过定义能力描述规范、工具集成接口与审计维度，推动多元主体在统一框架下互联互通，规避“各自为战”的治理盲区与合规风险。两者分别从工程稳定性与产业协同性，共同诠释驾驭工程作为核心管控框架与生态底座的价值内涵。

二、发展格局与生态现状

（一）技术发展应用现状

当前，驾驭工程正处于快速发展阶段，技术体系逐步完善，应用场景不断拓展，形成了“国外引领、国内跟进”的发展格局，核心技术聚焦于智能体协同、全生命周期治理、数据协同三大领域，应用已渗透至互联网、金融、制造等多个行业，展现出广阔的发展前景，与行业重点关注的人工智能、数字经济等领域高度契合。2026 年下半年以来驾驭工程进入开源基础设施爆发期，OpenAI 于 8 月全面开源 Codex Harness、DeepSeek 同期发布开源 Harness 框架，标志着驾驭工程从头部企业私有实践加速走向产业级公共基础设施，全球技术竞赛迈入“底座开源+生态共建”新阶段。

1.全球头部科技企业：围绕驾驭工程展开工具链与平台能力竞赛

从技术演进看，国际头部科技企业已形成相对清晰的工程范式。OpenAI、Anthropic 等企业率先提出驾驭工程理念并完成工业级规模验证与推广应用。Anthropic 自 2025 年 11 月起连续发布三轮工

程博客，形成从上下文工程、跨会话连续性到长程任务驾驭设计的完整技术演进链，推出 Claude Code 实现产品化落地。OpenAI 于 2026 年 8 月，采用 Apache-2.0 许可全面开源 Codex Harness，开放三层集成接口与递归智能体架构，为长程复杂任务的工程化分解提供了开源参考实现。同时，美国学术界和开源社区已围绕驾驭工程场景积累了大量规则模板与实践。欧盟、英国、加拿大等经济体依托完善的人工智能治理法规，将可信验证、算法审计与合规校验作为驾驭工程的刚性要求，形成以安全前置、全程可溯为特征的差异化发展路径。日本、韩国等国家则立足制造业、智慧城市等实体产业场景，强化生态协同、降低应用门槛，发展轻量化、高适配的工程化落地范式。

从产业生态看，驾驭工程已完成从零散企业级辅助工具到专门的智能原生软件工程体系的跃迁。商业端，OpenAI、Anthropic、微软形成面向开发者、企业服务的差异化产品矩阵，Salesforce、SAP 将驾驭层深度嵌入客户关系管理（CRM）、企业资源计划（ERP）业务底座，面向垂直行业交付一体化 Agent 解决方案。开源端，以 LangChain、AutoGPT、CrewAI、MetaGPT 等框架围绕编排、记忆持久化快速迭代，已形成较为成熟的可复用组件库。2026 年 8 月 Codex Harness 开源后，开源生态进一步从组件层面向公共底座升级，产业初现“商业产品矩阵+开源公共底座”双轮驱动雏形，开源底座降低了中小企业与垂直行业的接入门槛，商业产品则在安全合规、专属部署等能力上持续差异化。与此同时，云服务商通过合作

伙伴网络将开发工具与行业解决方案深度绑定，形成“平台+服务”一体化交付能力，共同为驾驭工程向更广泛行业渗透创造了条件。

表 1 国外驾驭工程实验/应用案例汇总表

时间	企业名称	应用场景	核心成效/实验文章
2025-11	Anthropic	Long-running agents 如何跨会话延续状态、保持方向。	发布文章《Effective harnesses for long-running agents》。
2026-02	OpenAI	智能体构建生产系统。	5 个月 0 手写生成 100 万行代码，3 名工程师处理 1500 个 PR。
2026-03	Anthropic	在真实应用开发里组织规划、生成、评估。	发布文章《Harness design for long-running application development》。
2026-03	LangChain	通用 Agent 底层编排驾驭框架、编程 Agent 基准调优实验。	编码智能体在 Terminal Bench2.0 从 52.8 分提升至 66.5 分，排名从第 30 位跃升至第 5 位。
2026-04	Anthropic	将 session/harness/sandbox 解耦成平台能力。	发布文章《Scaling Managed Agents: Decoupling the brain from the hands》。
2026-04	Stripe	代码提交与 PR 管理。	每周合并 1000+PR，部署失败率降低 85%。
2026-06	AWS	云托管生产级 Agent 运行时。	Bedrock AgentCore Harness 正式发布，全 AWS 区域可用，托管式 Agent 运行环境，把模型、工具、技能和指令通过配置定义。

（续）表 1 国外驾驭工程实验/应用案例汇总表

时间	企业名称	应用场景	核心成效/实验文章
2026-07	LangChain	LangGraph 专注于解决多 Agent 协作和长程任务的状态管理问题。	原生支持 Human-in-the-loop，允许流程在关键节点暂停等待人工审批或干预。
2026-08	OpenAI	Codex Harness 全面开源，构建开放智能体底座。	发布《Codex as a platform: build on the open agent harness》，Apache-2.0 许可开源，开放 CLI、codex exec 与 SDK 三层接口，支持递归智能体 Harness 架构。
2026-08	Microsoft	Microsoft Agent Framework Harness 与 Foundry Hosted Agents 正式发布。	从提供智能体构建 SDK 向提供治理化智能体运行平台的战略转移。

来源：中国信息通信研究院

2.国内发展现状：全栈基建与垂直工程化并行推进

我国驾驭工程技术发展与生态建设正处于快速成长期，呈现平台型企业全栈布局与垂直行业深度工程化实践并行的格局。

从技术演进看，国内以本土化适配、自主可控、行业合规为技术突破主线，同步追赶底层框架创新。阿里云、腾讯、字节、DeepSeek 等头部企业搭建自研驾驭中间层，针对信创环境优化权限管控、隐私计算与国产数据库适配，侧重共享记忆、流程柔性配置、多模型兼容等场景化突破。2026 年国内厂商在产品化与底层框

架两端持续发力，5 月阿里云将通义灵码升级为全栈智能研发助手 Qoder CN，支持多模型切换与 10 万+文件索引。8 月 DeepSeek 开源 DeepSeek Harness，采用 Cordis 微内核“一切皆插件”架构，24 小时内 GitHub 获约 5.5 万星。整体看，国内驾驭工程产品化规模化落地与开源底座自主可控并进，步入底层原创突破期。

从产业生态看，国内驾驭工程的应用渗透已从互联网头部企业向金融、制造、政务等垂直行业纵深拓展。中国信息通信研究院调研数据显示，截至 2026 年 5 月，企业内 AI 代码生成、自动化测试等关键环节工具渗透率分别达到 54.4%和 49.1%。驾驭工程供给侧形成平台自研与开源并行格局，平台型企业中，阿里 HiClaw、华为 AgentArts、腾讯 CodeBuddy/WorkBuddy 构建多智能体编排与长程驾驭能力，华为 AgentArts 长程任务完成率 80%、记忆召回 90%。开源底座中，DeepSeek Harness 采用插件化架构，填补国内通用驾驭框架开源空白。蚂蚁、海信、易鑫等垂直行业服务商在质量、安全合规细分赛道加速成长，产业正从单点工具与私有定制向平台化服务与开源共享并重演进。

表 2 国内驾驭工程实验/应用案例汇总表

时间	企业名称	应用场景	核心成效/实验文章
2026-02	字节跳动	开源项目 DeerFlow 2.0	通过将 sub-agents、memory 和 sandbox 组织在一起，配合可扩展 skills，让 Agent 能完成复杂任务。

（续）表 2 国内驾驭工程实验/应用案例汇总表

时间	企业名称	应用场景	核心成效/实验文章
2026-04	易鑫	汽车金融全链路（获客、风控、客服、资管等）AI 解决方案	基于 harness 治理体系，使单次任务持续执行 16 小时、跨 12 个会话连续推进、Agent 自主交付率 65%、运营效率提升 100%以上。
2026-05	阿里	HiClaw 企业级多 Agent 协作&Harness 落地解决方案	完整覆盖“Agent 协作一安全管控一数据支撑一持续进化”全流程。
2026-06	腾讯	WorkBuddy/CodeBuddy 全场景智能体矩阵+多智能体协同	覆盖办公/研发/设计场景，推出 AI Native Git 解决方案，使智能体作为数字员工常驻代码仓库，覆盖研发全流程。
2026-06	华为	云智果 AgentArts 企业级智能体平台	通过 Harness 框架支撑长程任务可靠运行、精准记忆管理，长程任务完成率达 80%、记忆召回准确率达 90%，支持万级会话安全运行。
2026-06	蚂蚁集团	阿福医疗 Agent 研发实践	EBPP 评测驱动驾驭体系，幻觉多层拦截校验，医生把关结果和阿福的一致率超过 90%。
2026-08	DeepSeek	开源 DeepSeek Harness（dsh），一切皆插件的 Agent 运行时底座	基于 Cordis 微内核架构，模型/工具/技能/会话/存储/循环/UI 全插件化，支持四种运行模式与子 Agent 调度，MIT 许可开源。

来源：中国信息通信研究院

（二）产业发展核心挑战

技术层面核心矛盾集中于三大瓶颈。其一，研发团队工程化思维缺位，叠加存量系统历史包袱。多数团队仍偏重大模型算法精度优化，忽视驾驭工程所需稳定性、兼容性、部署成本等综合工程指标，存量系统普遍存在参数冗余、环境适配差、容错不足等缺陷，致使全链路管控、软硬件协同迭代机制难以落地，技术原型向产业方案转化试错成本居高不下。Gartner 在《2025 年企业 AI 架构成熟度报告》中指出，AI 落地失败的根源并非单一技术缺陷，而是“资源供给、数据治理、系统协同、安全合规”构成的系统性架构失衡。**其二，行业工程体系高度碎片化，消解全域协同价值。**国内商用、开源大模型数量已超 200 款，主流模型厂商均采用私有应用程序接口（API）协议，接口参数、请求格式、流式响应规则、错误码体系无统一标准，形成严重的接口碎片化问题。据 2026 年 AI 应用开发行业调研数据显示，超 68%的开发者需为不同模型单独开发适配代码，35%的 AI 项目迭代耗时消耗在接口调试、格式兼容、异常适配环节，大幅提升开发与运维成本。**其三，全生命周期闭环管控普遍缺失。**传统静态线性开发模式与驾驭工程动态进化逻辑天然相悖，行业普遍重单点功能、轻体系治理，重短期交付、长效运维不足。多数企业仅零散试用局部工具，未搭建完整迭代闭环，无法动态处置模型漂移、数据衰减、推理异常等问题。中国信息通信研究院行业调研显示，68%企业将“AI 与工程融合不深入”列为首要落地障碍。

针对上述技术短板，可依托驾驭工程连接、编排、循环三大核心能力系统性破解。连接能力搭建统一能力目录与标准化协议，打通异构工具、数据、模型的互通壁垒，根治碎片化痛点。编排能力配套任务状态机、多层门控与异常恢复机制，保障长周期复杂任务稳定可控。循环能力依托隔离工作空间、多模式自动触发实现跨任务自主迭代，替代重复性人工操作。三类能力联动，将零散 AI 能力整合为可审计、可复用、可持续迭代的标准化工程底座，从底层化解适配不足、体系割裂、管控缺失三大技术难题。

标准化层面存在双重核心短板。一是顶层通用标准缺位，行业落地缺乏共识遵循。技术框架与落地准则，各类市场主体对其内涵、建设边界认知分歧较大，落地路径杂乱无序，难以规模化普及。量化评测体系缺失，缺少成熟度、工程质量、业务增益的量化考核指标，优质标杆难以筛选复制，行业迭代缺少明确指引。**二是分级分类落地标准空白，难以适配差异化需求。**不同行业、企业智能化底座、场景复杂度、合规要求差异明显，通用规范无法匹配个性化需求，落地普遍存在“一刀切”、适配度低等问题，大幅削弱落地价值。

针对上述标准短板，本报告自研 L1-L5 分级能力成熟度模型形成解决方案。该阶梯式分级体系按企业基础、风险等级划分差异化建设路径，精准适配各行业、不同规模主体，规避同质化落地弊端，同时以成熟度框架为核心底座，可逐层配套统一技术规范、落地实施指南与评测认证规则，补齐顶层标准、分类细则、量化评价

三重缺口，减少企业重复改造投入，形成可复用、可推广的标准化落地范式。

三、体系构建与实施指引

（一）驾驭工程的设计原则

作为智能原生时代应运而生的新型系统工程范式，驾驭工程的工程化落地与规范化实施，离不开系统性设计原则的引领与支撑。为破解 AI 与工程融合过程中的协同无序、效率不足、安全可控性欠缺等核心难题，确保其技术体系与实践应用的科学性、可行性及扩展性，结合学科发展规律与工程实践需求，驾驭工程的设计原则涵盖多个维度，本文重点阐述其中六项核心的设计原则，为驾驭工程的规模化落地提供指导遵循。

1.机器可读性优先原则

机器可读性优先原则是驾驭工程的基础性原则，核心是重构软件工程的信息表达范式，将软件系统从仅面向人类开发者的“经验型工件”，转变为 AI 智能体可自主解析、自主执行的“标准化工件”。工程师需要将约束条件、质量标准、验收准则等意图显式化且机器可执行化。通过消除隐式约定与认知模糊，降低生成过程中的推理不确定性，从而提升产出的一致性。为使 AI 智能体可靠地生产代码，所有原本隐含于开发者经验中的架构知识，都必须被显式化、结构化，并转化为机器可自主遍历的知识形态。代码仓库在该范式下转型为不再只是代码存储空间，还需嵌入架构决策记录、领域模型与模块契约，形成分层递进的信息架构。AI 代理应能从高阶架构文档进入，沿着可

计算的依赖关系逐步向下探索模块结构、领域实体及实现细节，实现自主上下文获取。在系统能力的暴露方式上，工程工具需将自身能力封装为机器可程序化消费的接口，而非仅面向人类的图形交互界面，为智能体端到端自主完成研发任务奠定基础。

2.渐进式信息披露原则

复杂系统的全量文档若一次性注入 AI 上下文，不仅浪费有限的注意力预算，更会加剧推理噪声与幻觉风险。**渐进式信息披露策略借鉴人类开发者的认知模式，主张将系统知识组织为可按需检索的层次化结构，使 AI 智能体能够沿着当前任务逐步获取恰好够用的上下文，恪守“最少载入”原则。**该策略将智能体的推理始终约束在其有效认知带宽之内，从而在保证生成准确性的同时，维持对大规模系统复杂度的驾驭能力。

3.边界约束与行为收敛原则

边界约束与行为收敛原则借助显式的边界定义，持续守护系统的概念完整性，使软件在长期演化中始终朝向预设的方向有序演进。传统开发流程依赖开发者的经验来避免架构问题，当 AI 成为代码的主要生产者时，必须在系统中显式地设置边界，通过架构约束、代码规范检查、依赖关系规则等手段，收敛 AI 的行动空间，从源头减少熵增。须将架构规则转化为系统可自动执行的常态化检查，清晰界定模块之间允许的依赖方向，禁止破坏性的跨层访问，并确保每一次交互都遵循预设的接口契约。该原则的本质，并非限制设计的可能性，而是通过主动收敛 AI 的求解空间，在多个代理高度并发的修改压力下，

从源头预防架构的缓慢腐蚀。

4.闭环反馈原则

闭环反馈原则通过多层次的自动化反馈网络构成自校正飞轮，推动系统持续演进，极大降低人工干预频次。须为 AI 智能体构建贯穿开发与运行全周期的多层反馈网络，进一步引入“智能体对智能体”的评审闭环，使其行为持续校订并向目标状态收敛。编译错误、测试失败、静态分析告警与性能回归等信号，应作为结构化事件实时回传至 AI 智能体，构成即时纠正的梯度信号。

5.安全合规内生原则

安全与合规内生原则核心是将安全管控与合规要求嵌入工程全流程，使之成为开发流程的原生属性，而非事后审计的补丁式附加。与“边界约束”原则侧重 AI 行为规范不同，本原则聚焦数据资产安全、生成代码安全及行业合规校验，要求在 AI 智能体协同全生命周期治理的每一个环节，均内置安全防护与合规检查机制，如 AI 生成代码时自动进行安全漏洞扫描、数据流转时执行脱敏处理、部署时校验行业合规标准。通过将安全隔离与合规校验作为独立的架构约束层嵌入框架，可将安全漏洞的逃逸率显著降低，从源头遏制风险外溢，为 AI 原生系统在受监管行业的安全部署提供了可复用的工程范式。

6.动态适配与记忆沉淀原则

动态适配原则要求驾驭工程不应将业务规则、协作流程与经验沉淀方式固化为系统内嵌逻辑，而应提供可配置、可扩展的机制框架，

使不同业务场景能够自定义治理策略。同时支撑智能体在执行过程中持续积累经验并优化行为模式。在业务治理维度，驾驭工程应提供“任务复盘触发”“经验沉淀提交”“关键节点评审”等通用管控原语，而非强制规定统一的执行细则。**记忆沉淀原则指多智能体协作时，记忆不应只停留在一次对话上下文中，应将相关注意事项与使用要求提炼为结构化的团队协作规范。**通过评审流程合入代码仓库，形成可供所有智能体消费的组织级共享记忆，用于降低重复试错成本、提升团队协作效率。智能体启动任务时，可先读取团队规范体系，明确前置检查项与协作约定，避免因信息不对称导致的反复失败。动态适配与记忆沉淀均应支持业务侧自定义配置，这一原则使驾驭工程从“固化流程的执行框架”升维为“支撑持续进化的适应系统”，为智能体在长期运行中积累知识、修正策略、形成稳定协作模式提供机制保障。

（二）驾驭工程核心技术体系

驾驭工程核心技术体系应围绕 AI 智能体的“可理解、可约束、可执行、可观测、可回放、可治理”建立统一能力框架。该体系的建设目的，是将大模型的一次性生成能力转化为可在工程边界内持续运行、可被组织管理、可接受质量验证的智能体执行能力。

报告将驾驭工程核心技术体系归纳为连接、编排、循环、效能、安全与评估优化六类能力，分层协同运作。基础层连接能力负责协议治理、资产接入，实现模型与外部系统语义互通，为上层能力提供底层支撑。执行层编排能力管控长任务状态、拆解任务并处理异常，循环能力依托工作空间隔离实现任务自动化推进、技能复用。保障层安

全治理能力把权限沙箱、全链路审计渗透所有执行节点，约束模型行为。进化层评估优化能力完成多维度评测，输出优化建议驱动全体系迭代。效能能力横向贯穿全部层级，通过上下文管理、模型路由与 Skill 复用，降低计算与人力成本。六类能力层层递进、相互支撑，共同构成驾驭工程的完整技术体系。

来源：中国信息通信研究院

图 4 驾驭工程核心技术体系架构

1.连接能力：模型与外部系统的语义互通机制

连接能力是驾驭工程的底层基础，其将模型意图转化为外部系统可执行的调用，并将外部系统返回的状态、结果、异常和反馈重新组织为模型可理解的上下文。大模型的输入和输出本质上是文本或结构化 token，而真实生产环境中存在数据库、文件系统、浏览器页面、终端环境、业务系统、工程工具、知识库、数据资产、其他智能体以及必要的物理环境接口。模型可以表达“查询订单”“修改代码”“调

用测试工具”“读取文档”“请求其他智能体协作”等意图，但这些表达本身不会直接触发现实系统动作。

一方面，连接层应建立统一能力封装机制。系统应将 API、Tool Calling、模型上下文协议（MCP）、智能体到智能体（A2A）、Connector、浏览器接口、终端接口、文件系统接口、数据库接口、应用系统接口等外部能力，抽象为模型可理解、可调用、可返回的语义接口。每一项能力均应明确名称、功能描述、输入输出结构、调用条件、权限等级、结果格式、错误类型和审计要求。通过统一封装，智能体不直接面对复杂异构系统，而是通过受控接口访问外部能力，从而降低集成复杂度和失控风险。

另一方面，连接层应具备能力发现与能力治理能力。系统不应允许工具和插件以不可管理的方式零散接入，而应建立统一能力目录，对工具、插件、Skill、Workflow、外部协议接口、数据资产和知识资产进行登记。符合要求的连接层，应能够回答“系统接入了哪些能力、谁可以调用、调用了什么、返回了什么、失败在哪里、是否越权”等问题。

来源：中国信息通信研究院

图 5 驾驭工程连接能力与外部系统语义互通机制

2.编排能力：长任务推进与过程控制机制

编排能力是驾驭工程从单轮问答走向复杂任务执行的核心。短任务中，模型的一次回答或一次工具调用即可接近用户预期结果，但当任务变成代码修复、需求拆解、测试执行、文档同步、运维诊断、数据分析、跨系统审批等长链路任务时，真正困难的不只是某一步能否完成，而是多次模型调用、工具调用、状态判断和结果校验如何被组织为同一个连续任务。

一方面，编排层应明确任务边界，记录任务目标、输入条件、约束规则、可用工具、权限范围、当前阶段、预期产物和完成条件。系统还应建立外部化过程状态和外部状态机（External State Machine），持续记录已完成步骤、失败路径、工具返回结果、已生成产物、仍然有效的约束、已发生的审批、待确认事项和异常状态。过程状态不能完全依赖模型上下文，因为上下文会被压缩、截断、替换或污染。

另一方面，编排层应支持下一步选择、异常恢复和结果验收机制。Workflow、有向无环图（DAG）、State Machine、ReAct Loop、多智能体协同等形态，均可视为下一步选择机制的不同实现。其中，多智能体协同正向子智能体递归编排方向深化，编排层应支持主智能体对长程复杂任务的递归分解与多并发子智能体协同调度，技术实证表明基于 Harness 的递归分解在任务处理效率与可靠性上优于单模型多轮推理。系统应支持暂停、重试、回滚、降级、路径切换、人工介入和任务终止，并通过测试用例、检查清单、静态分析、规则校验、外部反馈、预算限制、循环检测、超时控制和完成门控判断任务是否达到

验收条件。

3.循环能力：任务自动化持续推进机制

循环能力是驾驭工程从“人触发任务”走向“系统自动推进工作流”的关键能力层，其建设目标是将原本由人手动执行的“发现任务—分配任务—执行任务—检查结果—决定下一步”这一完整闭环，转化为可由系统自主运行的持续循环。

循环工程是驾驭工程在任务持续推进维度的自然延伸与能力增强。其建设围绕五个核心机制展开，第一，自动化触发机制。 系统应支持基于时间（定时任务、cron 调度）、基于事件（代码提交、CI 失败、Issue 创建）和基于目标（持续运行直至完成条件为真）三种触发模式。其中，基于目标的触发模式尤为重要，它不按时间触发，而是一直运行到预设标准真正完成为止，每完成一轮由独立评估者判断是否达标。第二，工作空间隔离机制。当多个智能体并行执行任务时，系统应为每个任务实例创建独立的工作目录，确保不同智能体的修改互不干扰，从根本上避免代码冲突问题。第三，技能与知识复用机制。系统应将团队代码规范、架构约束、历史经验等沉淀为可重复使用的 Skill 文档，使每一轮循环都能继承前一轮的经验，而非从零开始。第四，插件与连接器机制。 循环需要能够连接 GitHub、Slack、数据库、项目管理工具等外部系统，实现任务的自动发现与结果的自动输出。第五，子智能体分工机制。 不同智能体应扮演不同角色——探索方案、执行实现、验证结果，由独立的

评估智能体对执行智能体的输出进行校验，避免“自己给自己打分”所带来的评估偏差。

循环能力与编排能力、效能能力之间存在清晰的分工与协同关系。编排能力关注“单一任务内部”的步骤组织与状态管理，循环能力关注“跨任务”的发现、分配、执行、验证与决策闭环，效能能力中的 Skill 复用机制为循环提供可重复使用的知识资产。循环能力中的自动化触发则将效能资产的价值从“被动调用”升级为“主动运用”。三者共同构成了驾驭工程在任务推进维度的完整能力图谱。

循环能力的引入不应削弱驾驭工程的安全与治理要求。每一次循环的触发、执行、验证与状态变化，均应纳入驾驭工程的审计追踪体系。系统应支持循环的暂停、人工介入、条件终止和异常熔断，确保自动化循环始终处于可观测、可追溯、可干预的状态。

4.效能能力：上下文、记忆、复用与成本优化机制

效能能力通过确定性复用替代重复的人类指导与 GPU 推理，将稳定任务沉淀为 Skill、Workflow、脚本及缓存，降低人力、计算资源与系统成本。通过合理分工使大模型聚焦复杂规划与高风险决策，从而实现高质量、低成本的规模化智能应用。

一方面，效能层构建了确定性任务与复杂推理的分层执行机制。对于字段校验、固定标准作业程序（SOP）、数据搬运等可规则化的确定性任务，优先由脚本、轻量规则或普通计算资源执行，避免大模型从零规划。涉及跨领域判断、模糊语义理解或高风险决策时，调用

强模型能力。这种分工将 GPU 推理从海量低价值重复环节中释放出来，使计算资源精准投向真正需要开放推理与综合判断的核心任务，兼顾效率与质量。

另一方面，效能层通过多层技术机制实现能力的持续沉淀与优化。通过提示词缓存、结果缓存等机制减少重复计算，利用模型路由在强模型、轻量模型与固定规则之间动态分配任务。当某类任务形成稳定执行路径后，固化为 Skill、Workflow 或模板，模型仅需判断调用时机与适配参数，无需每次重新发明流程。该体系在保证可追溯与安全合规的前提下，显著降低重复解释、检索、推理与返工成本，驱动工程能力持续进化。

5.安全治理能力：权限、隔离、校验与审计机制

安全治理能力是智能体进入生产环境的核心前置条件，直接决定其落地可行性。驾驭工程在生产级治理实践中，呈现两大演进趋势。一是系统自动风控逐步替代人工审批，Anthropic Claude Code 自动模式危险命令拦截率达 89%，高于人工审批的 13.6%。二是权限管控从应用层规则下沉至操作系统内核，内核级 Agent 运行时由 OS 强制执行文件、网络、设备访问边界，安全治理的确定性与可靠性显著提升。

一方面，建立输入－输出－执行三侧联动的确定性安全边界。输入侧严格区分系统指令、用户指令与外部数据，从源头阻断提示词注入等攻击。输出侧通过模式校验、规则引擎与业务合规检查形成多重防线，过滤违规内容。执行侧强制实施最小权限原则，通过沙箱隔离、配额限制将智能体的操作范围锁定在预设边界内，从根本上降低安全

风险。

另一方面，将人工审批与全链路审计作为生产级治理的刚性要求。对于付款、批量数据导出、生产发布等高影响操作，系统强制触发人工审批，避免模型自主决策引发不可逆后果。同时将审计日志作为生产准入的硬性要求，而非事后补丁，确保能完整还原任务全流程，清晰地回答决策依据、权限来源、责任归属等核心问题，为安全事件的复盘与整改提供坚实基础。

6.评估优化能力：智能体运行质量的持续改进机制

驾驭工程的核心价值不止于完成系统的设计与构建，更在于基于智能体实际运行效果建立常态化的持续评估与优化机制。系统需构建覆盖执行能力、服务质量、资源效率、安全合规、业务价值五大维度的综合评价指标体系，其中执行能力聚焦任务完成效果，服务质量衡量用户体验，资源效率管控运行成本，安全合规守住风险底线，业务价值锚定最终产出，通过多维度量化评估全面反映智能体的真实运行状态。

一方面，多维度评估体系避免了单一指标导向的片面性。传统评估往往只关注任务完成率或响应速度，容易导致为了效率牺牲安全或质量。而驾驭工程的评估体系将技术指标与业务指标、成本指标与风险指标并重，既衡量智能体的技术执行能力，也评估其带来的实际业务价值和潜在安全风险，确保评估结果能够真实、全面地反映系统的运行成效。

另一方面，量化评估结果是驱动驾驭工程体系持续进化的核心依

据。通过对各维度指标的长期监测与对比分析，可以精准定位智能体运行中的短板，比如缓存命中率低导致的资源浪费、策略拒绝率过高影响的执行效率等。针对发现的问题，系统可以针对性地优化模型路由策略、沉淀可复用的工作流、完善安全合规规则，形成“运行－评估－优化－再运行”的闭环，推动驾驭工程能力的螺旋式上升。

（三）核心技术体系建设路线与关键工程原语

驾驭工程核心技术体系建设应坚持工程原语先行，避免单点功能堆叠的粗放式发展模式。工程原语是跨智能体应用、工具链与业务场景通用的基础能力单元，涵盖任务、会话、状态、工具、记忆、权限、评估、审计等核心对象，需明确其标准化定义与属性规范。关键工程原语应具备统一标识、版本管理、生命周期管控与审计追溯能力。以任务原语为例：给每一项自动化任务分配唯一编号 task_001、版本 v1.0、标准化定义待执行/执行中/已验收等状态，全链路可溯。企业需先梳理完整原语台账，再搭建平台，沉淀复用资产，依托统一底层标准实现多场景智能体一体化治理审计。

1.连接能力建设重点：协议治理、资产接入与边界控制

连接能力的建设重点，不是简单让模型调用更多工具，而是建立可治理的协议化接入体系。将代码库、CI/CD、数据库、工单等企业存量系统封装为可调用、可审计的标准化工程能力。

第一，统一多类协议治理。区分不同场景协议：Tool Calling 适配单模型工具调用、MCP 统一对外暴露数据源与业务系统、A2A 支撑多智能体任务协同、终端、浏览器接口打通物理运行环境。所有协议

纳入统一能力目录与权限管控。

第二，接口契约与调用边界。接入外部系统时，须明确输入输出字段、错误码、超时策略、幂等要求及敏感字段处理规则，调用边界需区分只读、写入、审批类与生产禁用类操作，对涉及个人信息、商业秘密、生产配置的接口设置更高权限与审计要求。

第三，校验结果可信度。外部返回数据可能存在过期、冲突或来源不明，系统应在回填模型前标注来源、时间、可信等级与权限状态，多工具返回冲突时需要求智能体标明冲突点，进入人工确认或规则校验流程。

第四，多智能体权限隔离。智能体间调用应遵循最小上下文与最小权限原则，仅传递必要任务信息，禁止默认共享完整会话或全部权限，被调用方输出作为受控观察结果回填，不得自动获取上级任务授权。

2.编排能力建设重点：任务管理、门控机制与异常恢复

编排能力的建设重点，是串联模型推理、工具调用、人工审批、结果校验，形成完整长任务流程。适配故障诊断、跨系统协同等复杂场景，维持稳定任务时序。

第一，任务状态管理。覆盖待执行、等待审批、失败、回滚等多类状态，状态变更全链路记录主体、输入输出与触发条件，长任务留存阶段目标、失败记录，规避上下文丢失引发的重复操作。

第二，多任务拆解。将复杂任务分为主任务（目标管理与验收）、子任务（具体执行）、检查任务（质量验证）与复盘任务（经验沉淀），

层级间通过结构化摘要交互而非无序共享上下文。

第三，门控机制。在关键节点设置质量与风险防线：输入门控判断目标明确性、权限充分性与上下文完整性，执行门控校验工具调用、文件写入与命令执行是否符合策略，结果门控验证产物是否通过测试、规则或人工验收，发布门控决定变更是否可进入生产。四层门控形成从任务启动到交付的逐级防护。

第四，异常恢复。覆盖工具故障、权限拦截、循环调用、审批超时等各类异常，匹配重试、降级、回滚、人工介入等处置方案，不可恢复异常需输出完整失败报告，标注故障节点、影响范围与修复条件。

3.循环能力建设重点：自主闭环构建与分层协同管控

循环能力的核心是将“单次任务编排”升级为“跨任务自主迭代”。使系统能够自动发现、分配、执行、验证任务并持续优化，打通从人触发到系统自动推进工作流的能力跃迁通道。

第一，搭建多模式触发与调度体系。 封装定时调度、事件监听、目标驱动三类触发引擎，统一分发 CI 异常、代码提交、工单事件。目标驱动循环依托评估智能体判定启停以规避无限循环，配套优先级队列，差异化配置研发、合规、运维任务权重，保障核心任务优先调度。

第二，隔离执行环境与资产复用。 依托全局 Worktree 为各循环实例分配独立目录，覆盖创建到销毁全生命周期，内置冲突检测阻断并行互斥修改，留存变更快照。循环自动加载 Skill、知识库与场景约束，同时归档优化规则入库，构建资产与循环双向迭代飞轮。

第三，受控协同与角色隔离。明确探索、实现、验证、评估四类智能体的分工契约，强制执行与评估角色物理隔离，独立评估智能体配备专属规则库，从机制上消除“给自己打分”的评估偏差。

第四，预算控制与熔断机制。覆盖时间、成本（Token/API）、迭代（最大轮次）三类预算，任一阈值触发自动暂停与告警。连续失败超阈值、单轮执行异常延长或检测到无效重复时自动熔断，通知人工介入排查，构成循环能力的安全阀。

4.效能能力建设重点：上下文预算、能力复用与成本治理

效能能力的核心是以可控成本维持高质量智能体运行，而非简单降低 token 消耗。智能体综合成本涵盖模型调用、人工干预、工具执行、等待时延、故障返工、安全治理六类，缺乏效能管控会导致系统成本、人工负担过高，无法规模化落地。

第一，上下文预算管理。系统应根据任务阶段、风险等级与模型能力设定上下文装配策略，划分强制输入、可选参考、按需检索、禁止输入四类数据边界，在保障推理判断完整性的同时精简冗余信息，降低计算资源开销与模型幻觉风险。

第二，模型路由策略。系统按任务复杂度、风险等级动态分配推理载体，高风险复杂任务调用强模型，标准化校验、摘要等交由轻量模型或脚本，完整记录路由策略并定期评估，平衡运行成本与输出质量。

第三，能力复用体系。将代码审查、用例生成等高频流程沉淀为

标准化 Skill、工作流与模板。成熟 Skill 封装验证过的业务规则、执行链路与验收标准，未经场景验证的临时提示词，禁止纳入组织共享资产库。

第四，缓存与成本治理。对系统提示、检索向量、通用模板分区缓存，配置时效与权限隔离，禁止跨租户复用。确定性校验、固定流程全部固化为版本化脚本资产，减少重复大模型推理，持续压降计算成本。

5.安全与评估建设重点：从风险阻断到可信运行

安全与评估能力需协同建设、深度联动，安全划定操作风险边界、落实风险阻断，评估校验任务执行质量与稳定性，二者结合才能让智能体从单点防护升级为全域可信运行。安全治理需贯穿任务全流程，摒弃单纯依赖提示词风控的粗放模式，将管控嵌入输入、执行、输出、复盘全链路，让模型运行始终处于确定性可控范围，这也是智能体落地生产环境的核心前提。

第一，输入阶段。评估关注任务目标清晰度、上下文充分性与外部资料可信度。安全关注指令注入、越权请求与恶意内容。系统应在执行前形成风险判断，必要时要求补充信息或进入审批。

第二，执行阶段。评估关注工具调用是否符合目标、执行路径是否有效。安全关注权限边界、沙箱隔离、敏感数据访问与高风险动作审批。工具结果、权限判断、审批记录应纳入统一事件流，使复盘能同时看到质量与安全问题。

第三，输出阶段。评估关注结果是否通过验收、是否存在幻觉与错误引用。安全关注敏感信息泄露与不当外发内容。关键结果须提供验证证据而非仅输出文本。

第四，持续优化阶段。评估样本与安全事件共同进入数据飞轮，失败任务优化上下文召回与评估规则，安全事件优化权限与审批策略，避免质量改进与风险处置相互割裂。

表 3 驾驭工程核心技术体系能力说明表

技术域	关键能力	主要成果物	验收重点
连接能力	工具/协议/系统接口封装、能力目录、双向语义转换	能力目录、接口说明、模式、调用日志	能力可查询、可授权、可禁用、可追踪
编排能力	任务边界、状态机、工作流、异常恢复、完成门控	任务状态模型、工作流模板、恢复策略	长任务可恢复、失败可定位、结果可验收
循环能力	自动化触发、工作空间隔离、独立评估、熔断机制	触发规则、工作树策略、评估模板、熔断配置	任务可自动发现与分发、每轮有独立评估、异常可熔断
效能能力	上下文管理、记忆/检索增强生成（RAG）、缓存、模型路由、Skill 复用	上下文规则、记忆规范、缓存策略、Skill 目录	减少重复推理和人工指导，复用率可统计
安全能力	输入隔离、输出校验、权限沙箱、人在回路、审计	权限矩阵、沙箱策略、审批记录、审计日志	高风险动作可审批，敏感访问可追溯

（续）表 3 驾驭工程核心技术体系能力说明表

技术域	关键能力	主要成果物	验收重点
评估优化	指标体系、测试回放、失败样本、持续优化	质量报表、回放集、优化记录	任务质量可度量，问题可复盘、可整改

来源：中国信息通信研究院

四、驾驭工程能力成熟度标准化分级

驾驭工程能力成熟度模型是本报告结合大模型与 AI 智能体驱动的智能原生软件工程特征与驾驭工程管控体系属性，自主设计的一套五级递进式评估框架。该模型借鉴了传统软件工程成熟度的分层评估理念，但对其评估维度进行了根本性重构，从面向人类开发团队的流程规范性评估，转向面向人机混合智能体系统的能力完备性与风险可控性评估。

模型设计的核心逻辑是从工具接入、运行底座、治理闭环、协同自治到规模生产五个递进等级，对应不同范围的连接、编排、循环、效能、安全、评估优化能力落地要求，每提升一个等级，智能体能够访问的系统、调用的工具、处理的数据和影响的业务范围同步扩大，风险控制能力同步增强 。以下将对 L1 至 L5 各等级的建设目标、必备能力、典型成果物、验收重点、风险控制要求及迭代治理要求进行系统阐述。

（一）L1-L5 能力成熟度模型

来源：中国信息通信研究院

图 6 驾驭工程能力成熟度模型能力覆盖矩阵

1.L1 工具接入级

L1 阶段依托基础连接能力建设，目标实现 AI、基础工程工具可登记、可调用、可关停，优先落地文档生成、代码检索、资料整理、测试用例草拟、日志摘要这类低风险场景。

L1 阶段的交付成果包含基础能力清单、工具调用记录、人工确认流程。应检查已搭建统一能力清单、留存基础调用日志，高风险能力默认关闭，能力异常可输出可读报错，使用者能清晰地查看智能体调用的能力项。进入 L2 前，应完成统一能力目录、基础日志搭建。

L1 阶段的主要风险是工具接入无序。企业应防止各团队绕过统一能力目录自行接入模型、插件和外部工具。该阶段应重点控制工具来源、调用权限、数据输入范围和基础日志记录，确保所有能力都可登记、可关闭、可追踪。

2.L2 运行底座级

L2 依托完整连接、基础编排能力搭建统一运行内核，保障智能体在多轮交互、多次工具调用、长任务执行时状态稳定。需实现会话管理、上下文装配、结果回填、超时管控、任务撤销、失败重试、上下文压缩、状态存储等功能。

L2 阶段的交付成果包含任务运行底座、会话存档、异常反馈机制、上下文压缩策略。应检查长会话运行稳定性，工具结果回填准确性，任务终止后资源释放效果，异常信息可读性与可追溯性。进入 L3 前，应完成权限分级和审计事件定义。

L2 阶段的主要风险是任务状态不稳定。企业应防止智能体在多轮会话中丢失原始目标、误用上下文、重复调用工具或在任务中断后无法恢复。该阶段应重点建设任务状态、上下文压缩、异常返回和资源释放机制，确保系统能够稳定承载长任务。

3.L3 治理闭环级

L3 依托连接、编排、基础安全三类能力，实现智能体行为可控、流程可审计、风险可阻断。搭建权限分级、安全沙箱、全链路审计、人工审批机制，针对文件写入、命令执行等高风险操作增设审批约束。

L3 阶段的交付成果包含权限矩阵、沙箱策略、审计与审批日志、异常处置流程。应检查权限覆盖范围、日志完整性、高危操作阻断、失败熔断能力。进入 L4 前，应完成工作流节点治理和任务回放机制。

L3 阶段的主要风险是权限和审计不完整。企业应防止智能体在未经审批的情况下执行写入、导出、命令执行、生产调用等高风险动

作。该阶段应重点建设权限矩阵、安全沙箱、审批流、审计日志和测试回放机制，确保高风险行为可阻断、可追溯。

4.L4 协同自治级

L4 依托连接、编排、循环、效能、安全能力，实现多智能体、多工具链、多工作流协同。在可观测、可审批、可回滚基础上提升任务自主执行水平，系统须具备任务拆解、子任务调度、工作流编排、跨系统调用、结果整合功能。

验收核验复杂任务拆分合并、故障节点定位、权限隔离、结果校验效果。搭建完成跨团队复用、统一治理策略后，便可升级至 L5。

L4 阶段的交付成果包含工作流目录、节点状态记录、审批节点、回滚方案、任务回放样本。应检查复杂任务拆分合并、故障节点定位、权限隔离、结果校验效果。进入 L5 前，应完成跨团队复用和统一治理策略。

L4 阶段的主要风险是多智能体协作混乱。企业应防止子任务权限污染、上下文无边界共享、结果合并不可解释和失败节点不可定位。该阶段应重点建设任务拆解、节点状态、角色边界、结果验收和局部回滚机制，确保协同自治仍然处于可观测、可审批、可恢复状态。

5.L5 规模生产级

L5 需落地驾驭工程六大核心技术能力，把驾驭工程打造为组织级智能原生软件工程基础设施。以循环工程实现从人机协同执行到机器自治进化的跃迁，搭建统一运行平台、能力市场、知识资产管理、安全合规、审计、质量评估体系，让智能体依托运行数据自主优化执

行策略。

L5 阶段的交付成果包含跨团队能力复用、质量量化评估、流程回归验证、多租户隔离、成本计量、统一安全管控、知识自动迭代。应检查系统能否支撑多用户、多业务线长期闭环自治运转。

L5 阶段的主要风险是平台规模化后的治理失效。企业应防止不同业务线重复建设能力、跨租户数据泄露、成本不可计量、安全策略不一致和知识资产失控。该阶段应重点建设统一能力市场、统一权限策略、统一审计体系、统一成本治理和统一运营机制。

来源：中国信息通信研究院

图 7 驾驭工程能力成熟度模型关键交付项

（二）L1-L5 能力成熟度的迭代治理要求

能力成熟度不是一次性评定，而应具备周期性复评和动态调整机制。随着模型能力、业务场景、工具体系和安全要求变化，原有等级结论可能不再适用。企业应定期检查能力目录、权限策略、测试样本、知识资产、成本指标和安全事件，确认系统是否仍符合当前等级要求。

对于能力扩展较快的场景，应建立变更评审机制。新增工具、开放写入权限、接入生产系统、引入多智能体协作或扩大外部用户范围，

均应触发等级影响评估。若新增能力突破原有风险边界，应先补齐相应治理能力，再扩大应用范围。

对于运行中发现的问题，应建立降级和整改机制。若系统出现连续越权尝试、审计缺失、测试回放失败、跨租户数据混淆、关键任务不可恢复等问题，应暂停相应等级能力，回退至较低风险模式，并形成整改计划。

（三）实施误区分析

基于上述 L1-L5 成熟度模型及分级风险控制与迭代治理要求，本章节进一步梳理企业在不同成熟度阶段推进驾驭工程时常见的偏差与误区。以下分析紧扣上述“能力可登记→运行可稳定→行为可治理→协同可自治→规模可生产”的递进逻辑，将常见实施误区归纳为认知定位、架构建设、过程控制、安全合规与规模化四个大类。

本节所列误区不应仅作为一般风险提示，而应作为建设评估中的负面清单。若系统存在对应问题，应在进入更高成熟度等级前完成整改，并形成整改记录、复测结果和责任归属。

来源：中国信息通信研究院

图 8 驾驭工程实施误区四分类

1.认知定位类误区

认知定位类误区主要表现为对驾驭工程的工程边界、方法论定位和建设目标理解不足，将其等同于提示词优化、上下文组织、模型能力展示或单点智能工具接入，割裂了驾驭工程统筹长流程、多主体、全链路治理的核心属性。

典型误区一是将驾驭工程等同于提示词工程。认为编写复杂边界约束提示词即可解决模型偏差、工具调用失败等问题，忽视了驾驭工程面向长任务、多工具协同的工程属性。提示词仅能优化单次问答的输出效果，无法替代标准化任务状态持久管理、分级细粒度权限控制、故障自动恢复、全链路操作审计追踪等底层运行底座核心能力。

典型误区二是重模型能力、轻运行底座。研发资源与考核导向过度倾斜大模型参数迭代、跑分指标优化，轻视长会话状态留存、分层上下文治理、断点故障恢复、资源隔离等基础支撑能力，最终出现演示场景效果优异，规模化落地频繁发生会话丢失、工具调用崩溃、流程中途中断等线上故障。

纠偏思路：将驾驭工程作为管控框架验收，以运行时内核、能力注册、上下文管理与全链路审计为准，不应仅凭模型回答效果或提示词模板判断驾驭工程建设成效。

2.架构建设类误区

架构建设类误区主要表现为重视工具数量和功能堆叠，忽视能力注册、协议治理、知识显性化、数据资产边界和生命周期管理，导致系统看似能力丰富，实际难以稳定复用和规模化治理。

典型误区一是盲目堆砌插件工具。一味接入各类 API 与外部系统，未搭建统一能力目录、调用规范、权限及版本管控，工具越多反而放大系统复杂度、权限风险与审计难度。

典型误区二是把记忆能力等同于聊天记录。无差别留存全部上下文，未管控记忆时效、可信度与访问范围，易造成过期信息复用、敏感泄漏、上下文污染问题，记忆与检索核心是按需调取合规可信知识，而非全盘存储。

典型误区三是忽视数据资产与知识资产边界。未区分数据资产的敏感等级与访问范围，也未管理知识资产的版本状态与可信等级，易导致越权访问、错误引用与合规风险。

纠偏思路：架构搭建以能力、知识、资产治理为根基，统一工具注册、接口契约、权限与下线流程，将业务规则、测试标准等沉淀为可校验复用知识，划分短期会话、长期记忆、知识库边界，杜绝无差别存储调用。

3.过程控制类误区

过程控制类误区主要表现为过度信任/不信任模型能力，忽视长任务执行过程中的任务边界、状态保持、异常恢复、结果验收和人机协同机制，导致智能体能够完成演示任务，却难以进入真实工程流程。

典型误区一是盲目追求全自动化，过早放开高危操作权限。部分企业在试点中过早开放代码写入、数据操作、生产发布等高风险动作，导致模型在上下文不完整或权限不清晰时执行关键操作，形成质量与安全风险。

典型误区二是过度控制，压制智能体自主能力。在流程中设置过多审批节点与人工确认环节，导致智能体每执行一步即被中断等待，长任务几乎无法自主推进。过度控制削弱了驾驭工程“边界内自治”的核心价值，使自动化带来的效率红利被频繁的人工干预抵消。

典型误区三是只看生成效率，不看质量闭环。部分实践以生成速度、代码行数为主要评价指标，忽视测试通过率、缺陷逃逸率与人工返工情况，易导致“效率提升、质量下降”，缺少评估复盘的自动化不具备生产价值。

纠偏思路：按照业务风险等级差异化配置操作权限，低风险任务全自动流转，高风险变更配置分级人工抽检，配套自动化缺陷复盘迭代机制。

4.安全合规与规模化类误区

安全合规与规模化类误区主要表现为企业落地优先级错位。试点阶段偏重功能效果演示，前置安全合规治理缺位，局部场景验证通过后直接全域复制推广，未统筹配套分级权限、数据隔离、全链路审计、成本管控、业务权责划分与长效运营治理体系。

典型误区一是先规模化推广，后补安全合规。优先铺开智能体应用范围，试点仅验证基础功能可用性。当智能体深度对接核心业务系统、代码仓库、线上生产环境后，单点安全缺陷会快速沿自动化链路全域扩散，后期重构整改、合规改造的人力与时间成本大幅抬升。

典型误区二是缺少审计追踪与责任边界。多环节操作无全链路记录，无法溯源任务发起人、数据访问、工具调用、结果确认主体，难

以满足企业合规要求。审计日志不应作为事后补丁，而应作为智能体进入生产环境的基础条件。

纠偏思路：应在试点阶段即建立最低安全基线，涵盖身份认证、权限分级、数据隔离、沙箱执行、敏感操作审批、输出校验与审计日志。进入规模化前，须具备统一权限策略、评估指标、运维监控与统一复盘机制，确保智能体能力可控扩展、可持续运营、可审计追踪。

五、驾驭工程赋能智能原生软件工程应用场景

（一）赋能智能原生软件工程关键节点与作业模式

驾驭工程对智能原生软件工程的赋能，首先体现为对软件研发全生命周期的系统性重塑。这种重塑并非在既有流程上叠加 AI 能力，而是围绕智能体作为核心参与主体这一根本变化，对需求、设计、编码、测试、运维各环节的作业模式进行再设计。

来源：中国信息通信研究院

图 9 驾驭工程能力赋能软件工程全生命周期映射

1.赋能需求分析场景

驾驭工程赋能需求分析环节实现从自然语言描述向机器可解析需求契约的自动化转化。针对传统软件需求分析的核心缺陷在于自然语言的语义模糊性与歧义性，非形式化的需求描述无法作为可验证的开发基准，导致后期需求变更与返工成本居高不下的痛点。驾驭工程通过意图解析、资料检索、需求结构化三大核心能力，构建需求规约的自动化生成体系。通过需求分析智能体与用户进行多轮结构化对话，主动识别并消解自然语言中的歧义点，补全缺失的前置条件与边界约束。平台内置的契约模板（如接口规范、行为描述语法）作为硬性格式约束，强制将业务描述转化为机器可解析的结构化数据与可执行测试脚本。该契约直接作为后续开发的完成标准，无需人工撰写传统需求文档。整个转换过程确保需求从“人的理解”变为“系统可断言”的客观规格，从源头消除因语义不清导致的返工。

2.赋能架构设计场景

驾驭工程赋能架构知识从隐性经验向显性可执行规则转化。针对传统架构设计高度依赖个体经验，架构知识以隐性形式存在于工程师头脑中，缺乏全局依赖分析与设计约束的强制执行机制，易导致架构一致性差与后期大规模重构的痛点。驾驭工程通过知识显性化、依赖分析、设计约束三大能力，实现架构设计的标准化与可复用。架构产出必须采用结构化模型表达，形成可被工具解析的依赖图谱与约束列表。设计约束（如模块隔离规则、通信方向、数据流转限制）被编码为自动校验规则，嵌入后续开发流水线，智能体会基于依赖图模拟故

障场景，提前识别设计薄弱点，最终输出一组可被下游各环节自动消费的架构规格，确保设计决策在编码、测试、部署中持续落地。

3.赋能编码实现场景

驾驭工程赋能编码环节实现沙箱隔离下的自主安全编码。针对编码阶段是安全漏洞与环境配置冲突的高发区，传统开发模式缺乏对编码过程的有效管控，代码质量与安全高度依赖开发者个人能力的痛点。驾驭工程通过代码检索、受控修改、影响分析三大能力，构建隔离式自主编码环境。采用隔离沙箱与静态规则相结合的自主编码机制。编码智能体接到任务后，平台在隔离环境中创建干净的沙箱，智能体在其中完成依赖引入、配置生成与增量编译。每次代码变更触发沙箱内嵌的代码检查器，捕获语法错误、安全风险等诊断信息，并将这些信息作为高优先级反馈注入智能体上下文。智能体在反馈驱动下自动发起多轮修复，直至代码通过所有规则检查与本地编译。通过校验的代码才被提交到代码仓库的审核队列。该机制保证主干代码不携带已知缺陷，同时隔离了外部环境的干扰。

4.赋能测试验证场景

驾驭工程赋能测试环节实现全流程自动化闭环。针对传统测试存在用例覆盖不全面、执行效率低、失败归因耗时等问题，难以满足快速迭代的开发需求的痛点。驾驭工程通过用例生成、测试执行、失败归因三大能力，实现测试全流程自动化。系统基于需求契约自动生成覆盖正常流程、异常场景与边界条件的全量测试用例，测试阶段启动后，多路并行派发测试智能体集群，在隔离沙箱中部署目标系统，模

拟用户操作与接口调用进行高强度探索测试。当测试过程中检测到故障，外部监控器立即终止异常进程防止错误扩散，状态重构智能体对原始错误信息进行精炼，剔除无关噪点，仅保留触发错误的操作序列、核心代码片段及运行时刻快照。随后将结构化错误信息指派给缺陷修复智能体，后者在新的沙箱中自动生成补丁并执行回归验证。形成“用例生成－并行执行－智能归因－自动修复－回归验证”自动流转的完整闭环。

5.赋能部署运维场景

驾驭工程赋能运维环节实现全链路风险管控与自动自愈。针对传统部署运维依赖大量人工操作，发布风险高、故障排查慢、业务恢复时间长，难以保障系统的高可用性的痛点。驾驭工程通过发布检查、故障诊断、回滚策略三大能力，构建主动防御与自动自愈的运维体系。发布检查环节，系统直接继承需求分析阶段生成的形式化契约与架构设计阶段的刚性约束，将运行态配置与契约中的接口承诺、性能基线、安全要求进行原子级比对，任何偏离都会立即阻断发布流程，并将偏差原因自动回填至需求阶段，形成从运维到需求的反向闭环。故障诊断环节，根因分析被严格锁定在“违反契约”“超出架构边界”等范围内，避免了无边界的盲目排查，大幅提升了定位效率与准确性。回滚策略环节，决策依据契约预定义的故障等级与架构依赖图，自动选择实例级、模块级或接口级的最小影响操作，而非一刀切的全量回滚。全流程轨迹沉淀为样本数据，持续优化规则。该机制将运维从“辅助建议”升级为“契约驱动、闭环自愈”。

（二）赋能智能原生软件工程质量与治理体系

驾驭工程对质量与治理体系的赋能，核心在于将分散、滞后的质量管控手段升级为前置化、全链路、数据驱动的内生治理能力。

1.赋能质量前置治理

传统软件质量治理依赖事后人工审核与流程卡点，规则执行弹性大、漏检率高，且治理成本随项目复杂度指数级上升。**驾驭工程将所有质量与治理要求（安全合规、架构标准、编码规范、质量指标）原子化嵌入全生命周期的形式化契约中，成为各阶段智能体不可逾越的执行边界。**从需求阶段的接口承诺、性能基线，到编码阶段的安全规则、依赖约束，再到部署阶段的发布标准，任何环节违反契约都会被系统自动阻断，实现“不符合治理要求的变更无法进入下一阶段”的硬管控，将治理从“人治”转变为“系统自治”。

2.赋能全链路可追溯

传统质量治理中，问题根因追溯困难，责任边界模糊，同类问题反复发生。**驾驭工程基于统一的契约标识与全流程操作日志，构建了从需求到线上运行的完整质量追溯链。**所有智能体的决策过程、代码修改记录、测试执行结果、发布操作步骤都被完整、不可篡改地记录下来，任何线上质量安全问题都可以精准追溯到原始需求偏差、架构设计缺陷、编码实现错误或发布流程违规等具体环节，并自动关联对应的责任主体与决策依据。所有发布检查、故障诊断、回滚决策的轨迹均被持久化存储，并与具体的契约版本、架构约束、代码变更关联，

安全审计人员可完整追溯每一次发布阻断的原因、每一次故障处理的决策链路，同时设置分层回退熔断机制，单模块故障局部回滚、全域偏差流水线整体熔断，避免局部缺陷通过自动化扩散为系统性故障。

3.赋能治理持续进化

传统治理体系依赖人工制定和更新规则，难以快速响应技术演进与业务变化，容易出现规则滞后或过度管控的问题。**驾驭工程通过持续改进阶段的运行评估、样本沉淀与能力优化闭环，实现治理体系的自主迭代。**系统自动采集全流程的质量与治理数据，分析规则的有效性与覆盖盲区，同时将历史质量问题与成功治理经验转化为结构化样本，反向训练各阶段的智能体模型与规则引擎。随着项目迭代，治理规则会自动优化，治理精度与效率持续提升，最终形成“数据沉淀－规则优化－质量提升”的正向循环。与此同时，多智能体协作时，应能够将记忆沉淀为团队共享规范，而非停留在一次对话上下文中，用于降低重复试错成本、提升团队协作效率。

（三）赋能多行业智能软件规模化落地应用

驾驭工程具备高度场景适配性，可根据不同行业的业务复杂度、监管要求与智能化基础，提供分层分类的落地方案，已在通用行业与重点垂直领域形成多类成熟应用场景，支撑智能软件规模化落地。

1.赋能通用行业场景

针对通用行业，建议实施标准化轻量化的规模化落地路径。通用办公、互联网服务、中小企业数字化等场景具备需求标准化、复杂度

低、成本敏感度高的特征，适配驾驭工程基础级与进阶级规范，以快速落地、低成本赋能为核心目标。第一，采用标准化组件复用、容器化轻量化部署、模板化流程配置的落地模式，依托成熟的自动化研发、智能运维与基础可信管控能力，快速搭建通用智能应用。第二，实施重点聚焦基础数据治理、标准化研发流程搭建与基础合规管控，优先保障系统稳定运行与高效迭代。第三，核心成效体现为研发效率提升、人力成本下降与基础智能化能力普及，大幅降低中小企业智能化转型门槛，为全产业规模化应用奠定基础。

2.赋能重点垂直行业场景

针对重点垂直行业，建议采用“通用能力底座+行业刚性校验”的落地模式。金融、工业、政务、医疗等关键行业具有场景复杂度高、安全合规要求严苛、需求个性化强的特点，通用层提供智能体编排、全链路可观测与权限管控等基础能力，行业定制层则需将各自领域的合规约束与专业逻辑固化为不可绕过的流水线卡点，实现驾驭框架下分行业精准匹配。

金融行业以合规风控、全链路可审计为核心，将合规约束从外部约束转化为系统内部刚性控制逻辑。驾驭工程设置双层卡点：前置规则引擎在模型调用工具前毫秒级拦截监管红线与交易限额，后置复核对授信、评级等关键输出强制二次校验，异常自动熔断并转入人工复核。例如，国内平台级金融科技软件即服务（SaaS）的服务商易鑫集团 Agentic Harness 三层体系可实现毫秒级合规熔断与业务全量可审计，覆盖获客、进件、风控、资金链路、智能客服等业务全链路，智

能 Agent 可持续执行 16 小时，自主交付占比 65%，整体运营效率提升 100%以上。

工业行业以研发资产确定性管控与代码全链路安全约束为核心，严控工业软件合规基线、AI 开发智能体权限围栏。工业软件 Harness 必须将行业规范、设备接口标准、代码安全阈值转化为不可绕过的硬约束，确保 AI 生成的工业代码、控制逻辑不突破工业软件兼容性与功能安全边界，同时，工业软件研发对版本一致性、变更容错校验提出更高要求。例如，家用电力器具制造商海信在 AI 原生开发实践中按高代码类、SAP 类、低代码类分类推进 AI 原生开发，搭建 Harness 工程体系，以代码仓为唯一事实源，通过前馈与反馈控制约束 AI 生成，优化 Spec 质量并重构研发流程。

政务行业以数据安全与全栈合规管控为核心，搭建跨部门智能体访问权限围栏。将数据分级、涉密管理、隐私脱敏规则固化为强制校验卡点，依托驾驭工程统一管控 AI 智能体数据调用、应用上线全流程，满足等保、密评、政务数据共享多重监管追溯要求。例如，广东省政务和数据局于 2026 年 4 月正式投入试运行省级政务智能中枢，该中枢通过聚合 3 大协同矩阵与 5 维能力基石，以测试、试运行、生产三区联运构建 Token 级安全沙箱，实现政务智能体从孵化到上线的全链路可控交付，现已归集超 100 个政务场景、适配十余款主流大模型，配套完整 Token 级安全防护体系。

医疗行业以诊疗精准性与患者隐私保护为核心，将器械合规、病历脱敏等嵌入校验链路。驾驭层统筹端云数据流转权限，约束原始患

者影像、病历数据不出院。例如，瑞金医院联合华为打造 RuiPath 病理大模型，通过端云协同混合计算架构，院内仅提取疑似肿瘤特征加密上传，节省 85%带宽的同时杜绝原始数据出域，云端全密态“可用不可见”完成推理增训，诊断结果脱敏后返回。流水线内置加密传输、全密态计算与输出审核，构成诊疗输出硬拦截门槛。结合数据飞轮持续验证，形成精准度越用越优、隐私与合规全程受控的工程化闭环。

3.差异化场景适配原则

针对不同应用场景，驾驭工程应根据企业规模、业务复杂度和服务对象差异进行分层适配。面向大型企业和复杂 2B 场景，应重点关注跨系统协同、租户隔离、权限分级、审计追踪和合规留痕。面向 2C 或轻量应用场景，应重点关注交互效率、结果稳定性、用户授权和数据最小化使用。面向金融、制造、政企等高风险场景，应将安全合规、可回放验证和人工审批作为应用前置条件。

针对跨行业生态融合与协同赋能机制，通过打通技术、场景与资源壁垒，构建跨行业、跨主体的生态协同体系。第一，实现技术资源跨行业复用，依托统一的标准化体系，共享智能研发工具、可信安全技术与数据治理方案，避免重复投入。第二，推动场景经验跨行业互通，将重点行业的成熟落地经验快速复制到新兴领域，加速全行业智能化进程。第三，建立产学研用协同机制，衔接科研院所技术创新、企业场景落地、高校人才培育与政府标准治理，形成“技术创新－产业落地－能力迭代”的正向循环，持续放大驾驭工程的产业赋能价值。

六、未来展望

驾驭工程概念的提出恰逢 AI 智能体从实验室走向生产环境的关键拐点。与过往许多 AI 概念不同，驾驭工程回应的是一个结构性而非周期性的工程需求，这一需求不随单一模型技术的迭代而消失，反而随智能体能力的提升而愈加迫切。

（一）从工程化组件向 AI 基础设施层演进

以更长远的视角观察，驾驭工程正处于从“上层工程化组件”向“AI 基础设施层”演进的轨道上。正如操作系统之于硬件、数据库之于应用系统，驾驭工程正在成为 AI 与现实世界之间不可或缺的“中间层”，提供的是通用大模型本身所不提供的可靠性、可控性与可审计性。结合产业节奏预判，预计未来 3 至 5 年，驾驭工程将经历三个阶段的发展。

第一阶段，概念验证与标准形成期（2026—2027 年）。行业将围绕核心定义、技术架构与成熟度模型达成初步共识。技术上，业界将围绕智能体核心工程能力开展验证，沉淀先行行业可复用实践。标准上，依托成熟度模型，国内机构加速标准编制，2027 年将落地首批行业规范。国际合规框架持续收紧，可控可审计成为全球共识，为本领域规模化发展扫清认知与规则障碍。

第二阶段，平台化与规模化期（2027—2028 年）。行业共识形成后，驾驭工程产业将迈入规模化发展阶段。头部企业推出集成编排引擎、安全沙箱、全链路观测、自动化回放等能力的成熟平台，形成标准化解决方案。应用场景从金融、互联网拓展至制造、政

务、医疗等领域，垂直行业差异化方案持续落地。同时开源工具不断涌现，降低企业落地门槛。配套细分标准、人才课程与分级认证体系逐步完善，推动驾驭工程从试点探索转化为企业通用标配。

第三阶段，基础设施化期（2028 年以后）。驾驭工程普及突破临界点后，将从产品形态升级为智能软件的底层基础设施，深度融入计算资源、数据、大模型底座，成为智能原生开发的默认支撑。在全球 AI 工程竞争格局下，欧美分别依托工具生态、合规体系构建优势，我国可凭借海量场景与完备产业体系，走出场景驱动的特色发展路径，最终成为保障智能软件规模化可靠运行的核心工程操作系统。

（二）人机协同化与新型开发者能力模型构建

随着驾驭工程规模化落地，人机协同将成为智能原生软件工程主流生产范式。传统开发者能力框架已难以适配产业变革，重构复合型人才能力体系、完善配套培育机制是行业长期发展的必然刚需，也将成为支撑 AI 工程产业持续扩张的核心内生动力。

生产层面，人机协同会形成标准化闭环生产机制，从根本上重构软件全流程作业模式。AI 工具将承接代码生成、测试编写、日志整理、基础运维等标准化重复工作，释放大量研发人力，人类工程师聚焦架构设计、风险研判、场景创新等高价值决策工作。依托驾驭工程搭建双向反馈闭环，机器输出成果、预警运行隐患，人工完成审核校准、优化管控规则，形成“机器提效、人工定策”稳定模式，能够大幅压缩项目交付周期，是全行业降本增效的核心抓手。

能力层面，开发者能力模型正经历结构性重构，单一编码或算法能力将被“四维复合能力”取代。Anthropic、Gartner、科锐国际、CSDN 联合调研数据显示，纯编码岗位需求锐减 47%，而具备 AI 协同能力的复合型开发者薪资溢价达 50%，岗位供需比 1:10。未来，行业将形成四维能力评价体系：一是复合技术能力，兼具软件工程、AI 模型工程、数据治理、可信安全与行业场景知识。二是人机协同驾驭能力，善用智能化工具，合理划分人机边界，管控智能体全生命周期。三是风险合规治理能力，掌握算法伦理、数据安全与行业监管，落实合规与风险预判。四是场景创新迭代能力，结合行业场景开展创新设计，具备工程优化、标准迭代与生态协同能力。

供给层面，分层长效人才培育体系加速落地，持续补齐产业复合型人才缺口。人才需求的结构性变化将传导至市场端，2026 年，传统软件开发需求整体已下降约 25%，但 AI 应用开发需求却增长 60%以上，供需缺口倒逼教育、产业、认证体系同步革新。需构建我国自主的驾驭工程领域人才能力认证体系，联合高校和培训机构开设驾驭工程与 AI 编程相关课程，推动软件工程专业课程体系改革，从“编码实现”向“AI 系统架构设计”转型。实施开发者培育计划，依托开源社区开展实战培训与认证。支持头部企业与高校共建驾驭工程实验室，培养掌握规则设计、行为治理与可观测性技术的复合型人才。

（三）生态协同与行业治理机制建设

未来产业竞争将从技术单点竞争转向生态体系竞争，需构建全球协同、国内规范、合规可控的生态治理体系，保障产业长效健康发展。

随着智能原生软件工程产业进入规模化、高质量发展新阶段，技术单点创新的红利逐步消退，产业竞争核心转向生态体系、标准体系、治理体系的综合竞争。

第一，产学研用深度联动，搭建分层互通的产业协同生态。创新端，科研院所聚焦基础理论研究、核心技术攻关、标准体系研制，夯实产业技术理论根基。产业端，企业聚焦场景落地、技术迭代、产品创新、工具研发，推动技术成果产业化落地。应用端，各行业用户反馈场景需求、优化体验，反向驱动技术与体系迭代。同时，搭建国家级、行业级智能软件工程创新平台与产业联盟，打通各主体协同壁垒，建立需求传导、技术转化、成果共享、经验互通的常态化协同机制，形成创新有方向、落地有支撑、迭代有反馈的良性生态闭环。

第二，全层级标准体系落地，形成统一可落地的产业行动准则。生态规模化扩张必然倒逼标准化建设提速，以驾驭工程 L1-L5 成熟度模型为起点，形成涵盖技术架构、数据流转、接口协议等的系列基础标准。随着开源 Harness 底座的发展，不同厂商 Harness 之间的能力描述与互操作协议尚未统一，亟需加快制定跨平台能力描述规范与互操作标准，实现组件互通、能力复用，同时为行业项目验收、合规核查提供客观依据，消除定制化开发带来的碎片化痛点，加速产业规模化复制。

第三，全周期长效治理机制成型，筑牢安全可控的产业发展底线。伴随智能体深度参与软件生产，算法风险、数据泄露、越权操作等隐患增多，常态化行业治理日益迫切。对外，2026 年 7 月国家网信办在

2026 世界人工智能大会上提出《智能体互信互联互操作全球合作倡议》，确立“互信、互联、互操作”三大支柱，倡导开放、非歧视、透明的国际标准制定原则。企业需落地权限隔离、操作审计、熔断回滚等配套制度，跨行业搭建风险共享台账，实现安全事件快速通报处置。内外协同，在开放合作中守住数据安全与算法合规底线，保障产业长期稳定健康发展。

CAICT 中国信通院

中国信息通信研究院 云计算与数字化研究所

地址：北京市海淀区花园北路 52 号

邮编：100191

电话：010-62301311

传真：010-62301311

网址：www.caict.ac.cn