arXiv:2603.06394v1 [cs.AI] 6 Mar 2026

自由对话，严格执行：面向灵活且可复现科学工作流的模式门控智能体人工智能

Joel Strickland[1*] Arjun Vijeta[1] ChrisMoores[2] Oliwia Bodek[2]Bogdan Nenchev[1] Thomas Whitehead[1] Charles Phillips[1]Karl Tassenberg[1] Gareth Conduit [1,3] Ben Pellegrini[1]

[1] Intellegens,The Studio, Chesterton Mill, 剑桥, CB4 3NP, 英国[2] 独立研究者[3] 卡文迪许实验室, J.J. Thomson Avenue, 剑桥, 英国

摘要

大语言模型（LLM）如今能够将研究者的自然语言目标转化为可执行计算，但科学工作流要求具备确定性、溯源能力和治理，而当大语言模型决定执行哪些内容时，这些要求难以得到保证。对来自十家工业研发利益相关者的十八位专家进行的半结构化访谈，揭示了两项相互竞争的要求——确定性约束执行，以及不受工作流僵化限制的对话灵活性——同时还指出任何解决方案都必须满足的边界属性（人在回路控制以及透明度/溯源）。我们提出将模式门控编排 作为解决原则：模式在组合工作流层级成为强制执行边界，因此，只有当完整操作（包括跨步骤依赖）通过机器可校验规范验证后，才会执行。

我们将这两项要求操作化为执行确定性（ED）与对话灵活性（CF），并以此为轴，沿验证范围谱系审视跨越五个架构类别的20个系统。评分通过多模型协议完成——覆盖三个LLM系列的15次独立会话——在模型间取得了从实质性到近乎完美的一致性（ED的Krippendorff's $\alpha = 0.80$ 与CF的 $\alpha = 0.98$ ），表明多模型LLM评分可作为人工专家小组的可复用替代方案，用于架构评估。

由此形成的格局揭示了一条经验帕累托前沿——没有任何受评系统同时实现高灵活性与高确定性——但在生成式与工作流中心化极端之间出现了一个收敛区域。我们认为，将对话权限与执行权限分离的模式门控架构有望解耦这一权衡，并提炼出三项操作原则——执行前澄清、受约束的计划–行动编排，以及从工具到工作流层级的门控——以指导采用。

1 引言

现代科学发现依赖于将多种工具串联起来的计算工作流，涵盖从数据准备到建模与分析的各个环节 [1–6]。随着证据不断积累，研究人员会更换模拟引擎、测试不同的特征化方法，并评估相互竞争的模型；这一进展具有组合性 [7–14]。然而，支撑这些工作的软件生态系统仍然支离破碎 [15, 16]——工具之间存在格式不匹配 [17, 18], 、参数语义不一致 [19, 20],以及不兼容的执行模型 [21]。因此，许多分析都是通过临时流程线拼装而成，参数选择、软件版本和执行环境也缺乏完善的文档记录 [22–24],这削弱了可复现性与可审计性 [25–34]。

*通讯作者：joel@intellegens.com

大语言模型（LLM）提供了一种对话式替代方案：用户用自然语言描述目标，系统则选择并执行工具 [35–41]。然而，提示驱动的代码生成会带来可复现性和溯源风险：临时实现可能在不同运行中产生差异 [42, 43], 相同的提示词也可能生成不同的算法或默认参数[44–46], 而由此造成的可追溯性缺失会削弱信任 [47]。提示注入、数据泄漏、未跟踪的供应链依赖等治理风险，也会使部署更加复杂；除非从设计之初就强制实施验证与可审计性 [48–51]。

传统的工作流管理系统（如 Galaxy [52], Snakemake [53], 和 Nextflow[54] ）通过要求显式的工作流规范，解决了其中许多问题。它们提供了强大的确定性、溯源能力和可扩展性保证，但也带来了较高的交互成本，并且对对话式或探索式使用的支持有限 [55, 56]。因此，当前的方法分化为两种趋势：一类是生成式、以代码为中心的系统，以牺牲可复现性为代价最大化灵活性；另一类是以工作流为中心的系统，以牺牲对话式使用为代价确保可复现性。

我们通过执行权限这一概念来阐释这种张力——即具体的可执行行为由谁负责，以及在计算运行（§3）之前必须满足哪些保证。与十家工业研发利益相关者合作开展的半结构化访谈（§2）显示出两项相互竞争的需求——执行确定性（需求A）和对话灵活性（需求B）——我们据此在 ED/CF 设计空间（§3.1）中绘制了20个代表性系统。该综述揭示了一个收敛区域；我们将其背后的原则称为模式门控编排，并在参考架构（§4）中将其具体化。

与既有工作的关系。最相关的框架是 Shin 等人的“(R)evolution”路线图 [41], ，该路线图从两个维度——智能（从静态到智能）和组合（从单体到群体）——描绘了从当前工作流系统迈向完全自主实验室的路径，并提出了一个自主科学的联邦式蓝图。我们的工作与之互补，但有三点不同：（i）我们依据实践者的需求（18 位专家、10 位利益相关者）而非技术优先的愿景来构建设计空间，由此发现，与自主化愿景相比，集成和确定性方面的顾虑更为突出；（ii）我们的 ED/CF 轴刻画了一种特定的权衡——对话权限与执行权限之间的耦合——而智能／组合维度并未涉及这一点（Shin 等人指出，生成式人工智能的非确定性与可复现性之间的矛盾是一项开放挑战，但没有将其转化为可操作的方案）；（iii）我们将模式门控编排确定为解决机制，这一模式并未出现在他们的蓝图中。他们的框架提出的问题是，系统应当具备多高的智能、采用何种组合方式；我们的框架则关注执行权限归属何处，以及计算运行前必须通过哪些验证。智能／组合矩阵描绘了工作流能力的长期演进；ED/CF 和模式门控则着眼于近期问题：如何让 LLM 驱动的系统在科学应用中值得信任。

贡献。

1. 对十位工业研发利益相关者的需求进行主题分析，揭示执行确定性（需求A）与对话灵活性（需求B）之间的张力，以及任何解决方案都必须满足的边界属性（§2）。2. 基于验证范围谱系，将20个代表性系统分为五组，并沿序数型 ED/CF 轴进行评分，在架构设计空间中加以可视化，从而揭示经验帕累托前沿（§3.1）。3. 一套可复用的多模型评分协议——由三个大语言模型系列开展的15次独立会话，并进行跨模型一致性分析——证明基于大语言模型的架构评估能够达到较高乃至近乎完美的评分者间信度（§3.1；附录C）。4. 将模式门控执行作为一种架构原则进行综合阐述，通过经模式验证的操作将对话权限与执行权限分离，并提出三项操作

原则：执行前澄清、受约束的规划—行动编排，以及从工具级到工作流级的门控（§4）。

5. 一种参考模式门控架构，具备经过验证的执行和端到端来源信息捕获；该架构作为一项架构提案提出，并指出其经验验证是近期最紧迫的优先事项（§5；§6；§6.5，第4项）。

2 用户研究与需求获取

为了解工业研发从业者对对话式人工智能工作流系统的需求，我们通过 Intellegens 的专业网络招募了来自十个利益相关方的 18 位专家，开展了半结构化访谈（通常每位利益相关方进行两次，每次~30 分钟；共 20 次访谈）。专家来自特种化学品、食品科学、材料与包装、半导体加工、建筑和制造业等领域（表 A.1）。访谈涵盖工作流程挑战、自动化机会以及对研发领域智能体人工智能的期望。访谈经同意后录音并转录。我们根据原始访谈记录，使用基于正则表达式的模式匹配，按照先验编码手册对主题进行编码，并排除访谈者和内部团队成员的发言。由于所有利益相关方均通过 Intellegens 的专业网络招募，样本属于便利样本，可能过度代表已在投资机器学习辅助工作流的组织；这限制了研究结果的可推广性，附录 A中对此有进一步讨论。完整的招募详情、保密协议、访谈结构及编码方案见附录 A。

2.1 主题景观

采用 Braun和Clarke 提出的系统性内容编码方法对回复进行编码 [57]，从 2,468 个发言轮次段落（18 位专家、10 位利益相关者）中的 1,135 个编码中归纳出 17 个主题。为确保提取均衡，每个主题均由固定的 20 个正则表达式模式定义——这是最小数量，能够让范围最窄的主题（例如渐进式自主性）达到充分的召回率。固定数量可防止词汇丰富的主题凭借词汇广度占据主导地位，但代价是可能限制稀疏主题的精确率。完整编码手册见附录 A；三部分敏感性分析（附录 B）——留一稳定性、跨会议收敛效度和自助法置信区间——证实了主题排名的稳健性，抽查精确率为 86–93%。

根据架构层面的影响，这 17 个主题分为四组（表 1）。确定性主导主题要求执行稳定、可复现且已验证；灵活性主导主题则要求采用对话式、由自然语言（NL）驱动的交互，以支持探索和快速迭代。三个跨领域主题描述的是从探索到执行 过渡阶段所预期的属性，而非与任一极相对应。范围外主题涉及部署和业务考量，将在 §6中简要讨论。

图 1 可视化了这一格局。 系统集成 是最突出的主题（191 次提及，10/10 利益相关者），其次是工作流与任务自动化 （112，10/10）和数据搜索与检索 （108，10/10）。灵活性主导主题实现了近乎普遍的广度，但提及强度较低。跨领域主题在两个轴上得分都很高，从而强化了其边界属性。

2.2 两项架构要求

这两极界定了一种在全部 10 位利益相关者中反复出现的张力。其各主题平均值（$\bar{x} = 88$ 为确定性， $\bar{x} = 75$ 为灵活性；表1）显著高于跨领域（$\bar{x} = 49$）和范围外（$\bar{x} = 41$）组别，这证实即使在按组别规模归一化之后，这两项关切仍主导了从业者话语。我们将它们表述为

表 1：按架构影响分组的从业者访谈主题。提及次数是所有专家的总计数；广度是出现该主题的利益相关者人数（10 人中）。组标题显示总提及次数和各主题平均值（$\bar{x}$），以便比较不同规模的组别。行颜色编码主题组：蓝色 =确定性一极，橙色 = 灵活性一极，绿色 = 跨领域，灰色 = 范围之外（与图 1一致）。

组	主题	提及广度	
确定性一极 （共 527 项； $\bar{x} = 88$）	系统集成	191	10/10
	工作流与任务自动化	112	10/10
	数据清洗与准备	70	10/10
	安全与知识产权保护	67	9/10
	领域／科学专业知识	62	9/10
	安全与伦理边界	25	9/10
灵活性一极 （总计 300； $\bar{x} = 75$）	数据搜索与检索	108	10/10
	自然语言界面	86	10/10
	可视化与报告	65	9/10
	多智能体／智能体间	41	9/10
跨领域 / 边界属性 （总计 146 项； $\bar{x} = 49$）	可解释性与透明度	83	9/10
	人工监督与控制	44	9/10
	渐进式自主性	19	5/10
超出架构范围 （共 162 项； $\bar{x} = 41$）	成本与效率	58	9/10
	采用与变更管理	51	10/10
	部署模型偏好	43	10/10
	数据聚合与匿名化	10	4/10

架构要求，因为它们独立于任何特定实现，约束了设计空间。

需求A。执行确定性。 源自六个确定性主导主题（共提及527次；所有10位利益相关者的首要主题均为该主题；表1，图 1）。从业者重视通过自然语言交互来表达意图，但期望对科学记录有贡献的计算保持稳定、可重复，并建立在定义明确的操作之上。几位利益相关者描述了渐进式信任模型：初始结果需经过人工验证，只有在系统表现出一致行为后，才会逐步放宽监督。一支化学团队明确区分了非正式探索（“玩聊天机器人”）与组织科学，并指出后者要求具备可重复、透明的流水线。

需求B。对话灵活性。 源自四个灵活性主导主题（共提及300次；利益相关者覆盖度为9–10/10；表 1，图 1）。从业者一致要求能够快速迭代——尝试替代方案、替换工具，并改进分析，而无需重写僵化的流水线。一位材料科学领域的利益相关者指出，大约四分之三的配方工作都涉及将已知组件以小幅变化重新组合，因此需要快速替换，而不是重新设计流水线。研究人员希望以对话方式描述目标，并让系统将意图转换为可执行步骤。

边界属性。 三个跨领域主题——可解释性与透明度（83次提及，10位利益相关者中有9位提及）、人在回路控制（44次，10位中有9位）和渐进式自主性

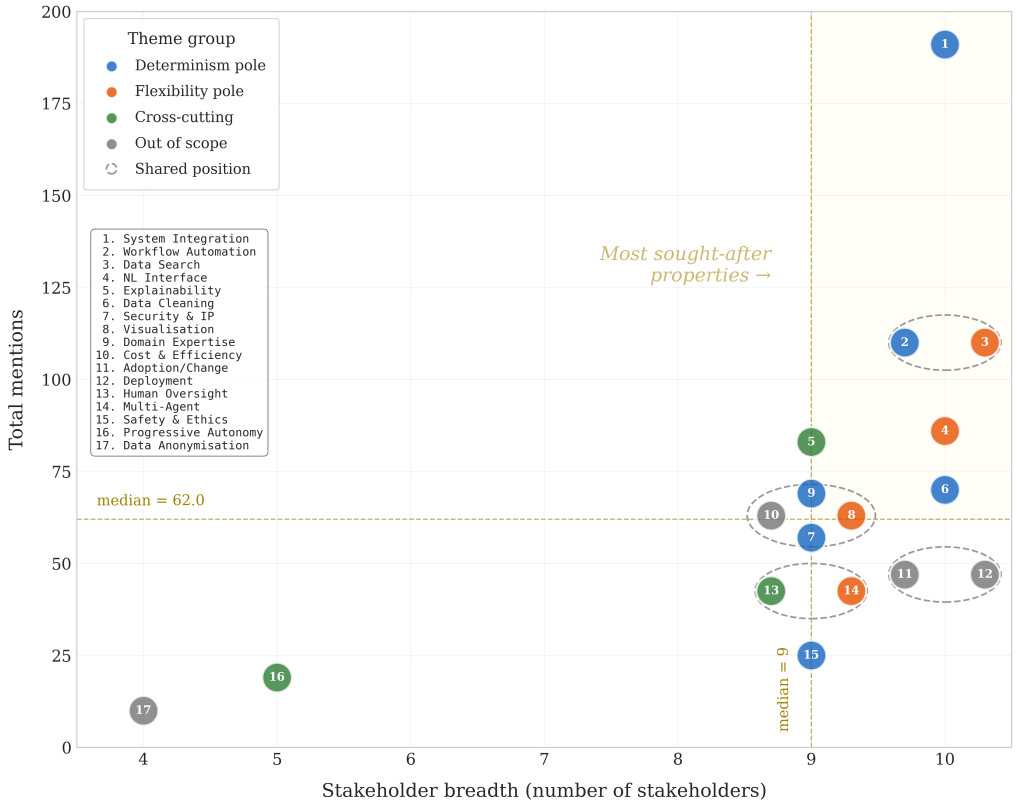

图1：从业者访谈呈现的主题景观。每个带编号的圆圈代表17个主题之一；图例将编号与主题名称对应起来。位置表示利益相关方广度（x轴：提及该主题的利益相关者数量，共10位）和总提及次数（y轴）。颜色表示主题组：蓝色表示 = 确定性一极，橙色表示 =灵活性一极，绿色表示 = 跨领域／边界属性，灰色表示 = 不属于架构范围。虚线标示利益相关方广度和总提及次数的中位数；阴影象限突出显示高于两个中位数的主题。点状椭圆圈出网格位置大致相同的主题。

（19次，10位中有5位；表 1）——既不是第三极，也不是可有可无的附加项。系统可以完全确定，却不提供任何人工审批步骤；也可以完全采用对话式交互，却完全不透明。当执行由显式、结构化的调用对象来中介时，审批关卡便轻而易举（调用本身就是可检查的产物），溯源信息也会在构建过程中自动记录（每次运行都是一条已验证、版本化的记录）。因此，这些都是边界属性——任何化解需求A/需求B张力的方案都必须满足的设计约束。

操作化。 我们将需求A操作化为 执行确定性 (ED)——即强制性执行前约束的力度，以及可供重放、经过版本化的执行制品的可用性；将需求B操作化为 对话灵活性 (CF)——即自然语言或智能体交互在多大程度上直接决定操作。现有系统是否也具备边界属性，取决于其架构承诺，并将在§6中进行考察。

3 对话式科学工作流的执行权限与架构范式

ED/CF 轴提供了一个用于比较各系统的框架，但应用它们需要一种视角，以区分不同系统如何介导从意图到计算的转变。我们

通过执行权限来刻画这一点：即决定贡献于科学记录的具体可执行行为的组件。在操作上，我们依据系统的最终执行产物来识别它——运行时接受为“要运行的内容”的最后一份机器可消费规范（例如，生成脚本、经模式验证的调用或工作流规范）。这不同于对话权限，后者管辖系统如何解释意图、提出操作并解释选择。

采用 ED 和 CF 两个维度（§2.2），本节从执行权限的角度审视 20 个代表性系统，并依据表 2 中的序数评分标准进行评分，沿验证范围谱系将其划分为五组，并在 ED/CF 设计空间中进行可视化。

3.1 审查范围与分类方案

表3 总结了 20 个代表性系统，包括商业产品、开源框架和典型工作流系统。系统的选择依据如下：(i) 覆盖全部五个验证范围组；(ii) 具备足以支持基于证据评分的公开文档；(iii) 拥有成熟的用户群或显著的社区采用度；(iv) 在各组内具有架构上迥异的特征（例如，Galaxy 和 Nextflow 在执行模型和社区规范方面有所不同）。本次选择旨在举例说明，并非 exhaustive；系统依据五项属性进行分类：可执行单元、最终执行产物、执行前验证、来源信息捕获以及主要执行权限。我们将通用大语言模型聊天的“通用用法”作为基线纳入。每个系统还会获得 ED 和 CF 序数评分（评分标准见表2）；由此生成的设计空间图（图 2）将在 §3.2中讨论。

根据这些属性，每个系统会沿着验证范围谱系被划分为五组（反映典型/默认用法；具体部署可能有所不同）：**生成式**（没有正式的执行前门控）、**工具增强**（仅进行工具级验证）、**模式门控**（对话优先，并强制进行模式级验证）、**工作流 + 自然语言**（工作流优先，采用自然语言（NL）接口），以及**以工作流为中心**（具有显式工作流产物，并进行强静态验证）。表3 中的行颜色表示这些分组；该表还使用了标准缩写 VCS（版本控制系统）和 CI（持续集成）。

设计空间图的序数评分（ED/CF）。 每个系统使用评分标准被赋予序数（1–5）执行确定性与对话灵活性分数，见表2，并可视化于图2。

表2: 执行确定性（ED）与对话灵活性（CF）的序数评分标准 .

评分	ED（执行确定性）	CF（对话灵活性）
1	不受约束的生成代码；没有正式的执行前执行门控	没有对话式界面（仅限 DSL/配置/图形用户界面）
2	主要依赖运行时检查；执行前约束较弱约束条件	有限的自然语言辅助（表单 帮助、搜索，提示）；自然语言不控制执行
3	开发者定义的工具接口；部分验证-但并非统一受模式约束	自然语言辅助编写/修改工作流；执-执行需要一个外部化制品
4	经模式/类型验证的强工具调用执行前验证	NL 直接 选择/参数化 工具/工作流 调用 via 结构化 调用
5	显式工作流规范/注册表，带静态验-证；构造性可复现	自由形式自然语言或智能体循环直接驱动仅受最少结构约束的操作

表3：综述系统与架构属性（标识符对应图2）。行颜色表示范式组别：红色 = 生成式，橙色 = 工具增强，绿色 = 模式门控，紫色 =工作流 + 自然语言，蓝色 = 以工作流为中心。ED/CF 分数见图2 及附录C。

ID	系统	领域	接口	可执行项	验证	溯源
1	大语言模型聊天（通用）	通用	聊天（自由形式）	生成的代码	无/非正式	临时日志
2	*GitHub Copilot*	软件 dev	内联提示	代码片段	None	外部（版本控制系统/持续集成）
3	*AutoGPT* 风格 智能体	通用	自主 智能体	计划 + 代码 + 工具	仅运行时	部分日志
4	*ReAct* 风格 框架	通用	推理-行动循环	计划 + tool 操作	工具检查	部分日志
5	*LangChain*	LLM 框架	聊天 + 智能体	代码定义工具	有限检查	部分 (观测/日志)
6	多智能体科学	发现	对话智能体	工具图/计划	部分检查	部分日志
7	语义内核	LLM 框架	聊天 + 插件	工具调用	部分 (dev 类型)	部分（遥测）
8	*OpenAI* 助手 / 工具调用	通用	聊天 + 工具调用	工具调用	若使用模式则较强	部分（应用日志）
9	*Copilot Studio / Power Automate*	企业 auto	聊天 +低代码	连接器 + 流程	强（合约）	显式 (run 历史。)
10	*n8n*	自动化	低代码（+自然语言）	节点工作流	强 （节点模式)	显式（运行）
11	自然语言辅助工作流 编写	Sci 工作流	自然语言辅助	生成 → 已验证 wf	强 （验证后）	显式 （之后实验）
12	*Dataiku DSS*	分析/机器学习图形用		配方/工作流	强（类型化）	显式（血缘）
13	*Galaxy*	生物信息学	表单（有限的自然语言）	工作流（DAG）	强	显式（历史记录）
14	*Snakemake*	Sci 工作流	DSL/配置	工作流（DAG）	强	显式 文件/日志
15	*Nextflow*	Sci 工作流	DSL/配置	工作流（DAG）	强	显式（运行/元数据）
16	*nf-core*	生物信息学	模板	精选工作流	强 （持续集成+有效。）	显式（版本化）
17	工作流注册表	多领域注册中		已注册 工作流	强（策略）	显式（版本化）
18	*AWS Step* 功能	云工作流	领域特定语言 + 控制台	状态机工作流	强（静态）	显式（历史记录）
19	*Apache Airflow*	数据/机器学习	Python 有向无环图 + UI	DAGs	强 (DAG 解析)	显式 （数据库+日志）
20	*Kubeflow / Argo*	机器学习 工作流	UI + YAML/软件开发工具包	流水线（有向无环图）	强（规范）	显式 （产物）

评分协议。针对 20 个系统进行了评估，在三个模型系列——ChatGPT 5.2、Claude Sonnet 4.6 和 Gemini 3.1 Pro——中，每个系统均在全新的对话上下文中使用相同的结构化提示进行 15 次独立评分会话。在全部 15 次运行中，Krippendorff'sα 的 ED 为 0.80，CF 为 0.98，表明模型间一致性达到较高至近乎完美的水平。图 2 采用 15 次运行的均值，以区分中位位置相同的系统。各模型系列的一致性统计、完整提示模板、逐会话评分以及模型间一致性分析详见附录C。

ED 描述了"实际运行的内容"在多大程度上受到已验证、可重放工件的约束，且不受对话变异性的影响。它并不保证科学正确性，也不能消除各个工具内部的非确定性（例如随机算法）。经审查的系统在 ED/CF 空间中勾勒出一条帕累托前沿（图2）：没有任何系统能在两个维度上同时超越这条前沿。这条前沿是经验性的，但也部分源于结构，因为授予自然语言直接执行权限的系统，按其架构设计，往往会放弃执行前验证（§6.5）。评分表示用于比较性可视化的序数位置，而非定量性能测量。

3.2 架构设计空间

图2 展示了经审查系统在 ED/CF 空间中的分布。这些维度体现的是系统架构的属性，而非语言模型的属性；根据执行的中介方式，同一模型可以位于任意位置。五个组之间的空间分布描绘出一条由经验帕累托前沿所体现的非线性权衡关系。

图中的虚线 2 描绘了这一前沿——由连接非支配位置的分段线段构成——展示了给定对话灵活性水平下观测到的最高执行确定性（反之亦然）。由于底层得分是序数数据，因此采用直线插值，以避免暗示存在连续的函数关系。

ED/CF 轴描述的是 从业者的需求——交互完全灵活 且执行完全确定——而非软件的内在属性。因此，（5，5）的理想状态代表一种需求侧吸引子：所有系统都面临着向其靠拢的压力。随着时间推移，生成式工具将增加验证和溯源能力，而以工作流为中心的工具将加入对话式接口，因此，图2应被视为对不断变化的格局所做的一次有时效性的快照。

我们将模式门控区 定义为执行确定性和对话灵活性均达到或超过 3.5 的区域。在审查的 20 个系统中，只有编号 8 和 9 位于该区域——它们都是智能体系统，会在每次工具调用时强制进行模式验证。工作流 +自然语言组（编号 10–12）具有较强的执行确定性（≥ 3.7），但对话灵活性不足（$= 3.0$）；而工具增强组（编号 5–7）缺少强制验证，因而无法使执行确定性超过阈值。两组都在向该区域靠拢，但尚未到达。

尽管分类体系识别出五个组，但从架构上看，它们可归为三种范式。生成式组和工具增强组具有共同特征——大语言模型保留直接执行权限——因此合并考察，归为**范式 I** (§3.3)。以工作流为中心组和以工作流+ 自然语言组均将执行权限交由显式工作流规范： **范式 II**(§3.4)。模式门控组（标识符 8–9）构成**范式 III** (§3.5)：这些系统已在工具调用层级实现了这一模式；我们提出，将门控扩展至组合工作流层级，有望同时提升执行确定性和对话灵活性。

3.3 范式 I：生成式代码中心方法

第一类方法赋予对话式推理对可执行行为的直接控制权：大语言模型将提示词转化为代码、脚本、笔记本单元格或 Shell 命令（表 3，标识符 1–4；图 2）。一类与之密切相关的**工具增强**系统（标识符 5–7）则增加由开发者定义的工具接口——函数调用 API、插件注册表或基于协议的工具服务器（例如模型上下文协议，MCP [58]）——这些接口会根据模式验证单次调用，但验证仍局限于单次调用。这些系统以形式化换取表达能力，适用于探索性分析和快速原型开发；然而，由于对话权限与执行权限相耦合，生成的代码往往具有临时性，运行结果可能各不相同，验证也较为非正式。广泛采用的智能体代码助手——GitHub Copilot、Cursor、Claude Code——具备复杂的内部规划和执行追踪能力，但这些能力对用户而言大多仍不透明；对于科学研发，透明度、可审计性和人在回路控制必须成为一等的面向用户功能。

架构特征。 执行权限位于大语言模型/智能体（或其控制器）中，最终执行产物通常是生成的代码或约束较松的工具操作。意图与执行之间的契约主要是对话式的，而非可由机器校验的。

3.4 范式 II：以工作流为中心的系统

第二个区域由这样一类系统构成：它们通过将计算外化为显式的工作流规范来优先保证执行确定性——领域特定语言（领域特定语言）、配置

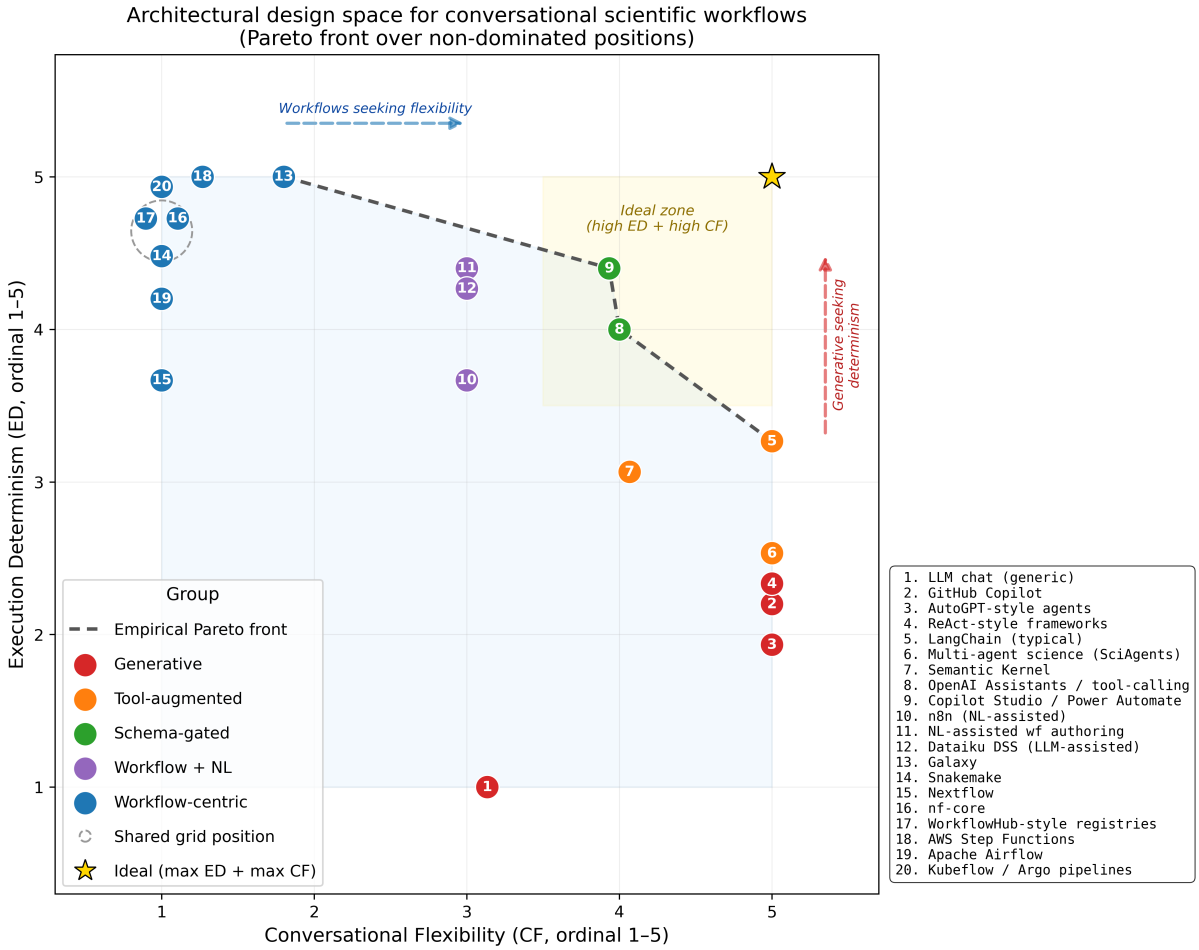

图2：对话式科学工作流的架构设计空间。系统根据 15 次独立评分运行中 ED 和 CF 的平均分进行定位（附录 C报告共识中位数；此处使用平均值，以便在视觉上区分位置重合的系统）；点标签对应系统 ID。颜色表示五个组别：生成式（红色）、工具增强（橙色）、模式门控（绿色）、工作流 + NL（紫色）和以工作流为中心（蓝色）。虚线是经验帕累托前沿，由分段线性线段连接非支配位置构成（即观测到的帕累托最优点之间的直线）；前沿下方的阴影区域包含受支配位置。金色星形标记表示高灵活性和高确定性的理想区域。 注意：分数为序数尺度（1–5），并非定量值；此处绘制分段线性前沿是为了直观呈现权衡形态，不应将其解读为位置之间存在区间尺度距离或连续函数关系。

文件，或用于定义工作流有向无环图（有向无环图）、数据依赖关系和参数化的图形化编辑器（表 3，标识符 13–20；图 2）。它们提供强可复现性、溯源和可扩展性保证，适用于稳定的流水线与协作复用。主要局限是交互成本：用户必须学习工作流抽象和配置语法，而迭代实验往往需要手动编辑规范。

一组密切相关的**工作流 + 自然语言**平台（标识符 10–12）通过自然语言接口扩展了这一模式，例如对话式创作、自然语言到工作流转换器或基于聊天的参数补全，同时仍将已验证的工作流规范作为执行权限的依据。它们具有较强的执行确定性（≥ 3.7），但对话灵活性有限（$= 3.0$），因此位于模式门控区之外（§3.2）。

架构特征。执行权限归属于工作流规范（或经整理的工作流注册表/执行器），最终执行产物是显式的工作流实例（有向无环图/领域特定语言/配置），由运行时进行验证和执行。自然语言可以辅助人工编写，但不控制执行。

3.5 范式 III：模式门控工具执行

第三区域包含模式门控组（表3，编号 8–9；图2）：对话优先的智能体系统，会对每次工具调用强制执行模式验证。与生成式系统（允许大语言模型自由执行）或以工作流为中心的系统（要求用户编写规范）不同，这类系统在保留完整对话式交互的同时，通过已验证的结构化产物来调控执行。

在审查的 20 个系统中，编号 8 和 9 最接近理想位置 (5, 5)，是仅有的位于 §3.2所定义模式门控区内的系统。它们之所以处于该区域，是因为每次工具调用都必须经过网关式模式验证：

- **OpenAI 助手 / 工具调用（ID 8）**。在 `strict: true` 模式下，Assistants API 通过约束解码提供强模式级验证，在分发前拒绝不符合开发者定义的 JSON Schema 的函数调用参数集。大语言模型可以自由推理要调用哪个工具以及如何填写其参数（对话灵活性 $= 4$），但只有通过模式门验证后，工具才会执行（执行确定性 $= 4$）。
- **Copilot Studio / Power Automate（编号 9）**。连接器通过 OpenAPI规范定义，包含类型化参数和操作契约；Power Automate 的 Flow Checker 会在执行前根据连接器模式验证流程，每次运行都会生成一条显式的历史记录。生成式编排层会根据用户消息选择并排列主题、工具和智能体（对话灵活性 $= 4.4$），但每次调用连接器都必须满足其合约（执行确定性 $= 3.9$）。

在这两种情况下，验证都是强制且排他的：不存在绕过模式边界的执行路径。这一特性使这些系统有别于工具增强系统（编号 5–7）：后者提供开发者定义的工具接口，却允许大语言模型通过推理采取未经验证的操作。对单个工具调用进行模式验证已十分普遍——OpenAI 函数调用、Anthropic工具使用和 MCP 都为单次调用定义了类型化接口——但编号 8 和 9 更进一步，将模式设为执行的唯一途径。

架构特征。 执行权限位于经模式验证的工具/工作流层；最终执行产物是一个完全解析且机器可校验的调用对象。表 4总结了范式差异。第 4–5 节将底层原则形式化，并将其从工具级门控扩展到组合工作流编排。

4 作为设计原则的模式门控编排

模式门控组中的两个系统（标识符 8–9，§3.5）具有一种值得形式化的架构承诺。我们使用 *模式门控* 来表示：**运行时拒绝执行任何无法表示为版本化工具或工作流调用、且无法在边界处通过机器可校验模式验证的拟议操作**。对话可以塑造意图，但不能绕过验证。这使权限分离决策变得明确：

- **对话权限：** 解释意图、提出候选动作、提出澄清问题并解释选择。
- **执行权限：** 仅通过满足显式、机器可读约束条件的操作来选择并运行计算。

表 4：对话式科学工作流的架构范式（摘要）。列颜色对应表 3 中定义的范式组。

维度	生成式	以工作流为中心	模式门控
主要执行 *au-* 权限	大语言模型/智能体	工作流规范/注册表	经模式验证的工具/工作流层
自然语言的作用	生成可执行项制品	协助编写（有限）	表达 意图； re-求解参数；提出操作
可执行单元	生成的脚本/单元格/命令	显式 有向无环图/领域特定语言/配置	已验证的工具调用和工作流
执行前验证 *tion*	极少或不存在	强	强（模式/类型/约束检查）
确定性 跨越 *runs*	无法保证	High	高（在已验证的范围内）
来源信息捕获	隐式／临时性的	显式	显式 （工具级 and 工作流级）
迭代式探索	高，但不安全	有限／开销高	高， 在受治理的约束条件
典型故障模式	幻觉 API，静默变异	配置 摩擦，刚性	覆盖 限制 of 模式／工具注册表

在实际操作中，这使“运行了什么”成为一个显式对象——工具／工作流标识符（及版本）加上已解析参数集——而不是隐式的提示结果。使用模式来描述工具接口、却允许未经验证的操作的系统，只是由模式描述的系统；模式门控系统则将模式作为执行的唯一途径。

编号 8 和 9 已在单个工具调用层级强制执行这一不变量。然而，科学工作流通常会串联多个步骤，并包含单次调用验证无法捕捉的跨步骤依赖。以材料发现流程为例：(i) 加载数据集，(ii) 使用该数据集中的列训练代理模型，(iii) 使用训练好的模型运行逆向设计。工具层级的验证可以确认每次调用单独来看都符合类型要求，却无法检测步骤 (ii) 中的目标列是否不存在于步骤 (i) 的数据集中，也无法检测步骤 (ii) 中训练的模型所基于的属性是否与步骤 (iii) 的要求不同。这些是跨步骤类型和依赖错误：每次调用在局部都有效，但组合计划并不可靠。工作流级模式门控会在任何计算运行之前，通过验证整个有向无环图中的步骤间数据流类型、依赖顺序和参数兼容性，捕捉此类不匹配。

为了满足科学实践所要求的确定性（执行确定性 = 5），模式门必须从单个工具调用扩展到组合工作流层级，在任何步骤执行之前，对整个计划中的结构、依赖关系和类型约束进行强制检查。我们将工具层级的模式称为模式门控工具执行，将工作流层级的扩展称为模式门控编排。

4.1 操作原则

模式门控不变量——未经模式验证，任何内容都不得执行——带来了操作层面的影响，使其区别于上文评述的无约束智能体系统和僵化工作流系统。

执行前澄清。由于模式门会拒绝不完整或类型错误的调用，缺失字段、类型不匹配和约束违反会以对话提示的形式显现——将无声失败转变为结构化协商。调用对象也是审批关卡审查并由溯源记录捕获的产物，因此边界属性（§2.2）是架构层面的必然结果，而非后续附加的功能。

受约束的规划—行动编排。 模式门控编排将推理与执行分离。在**规划模式**下，模型充当完全自主的智能体推理器——解读意图、分解任务，并通过开放式对话补全参数 [38]。在**行动模式**下，模式门控不变式开始生效：未经验证，任何内容都不会执行。无约束的智能体可以跳过验证，依据近似推理采取行动；模式门控系统则不能。这种双模式结构使规划阶段的高度对话灵活性与行动阶段的高度执行确定性得以并存。

从工具级门控到工作流级门控。 工具级模式验证——OpenAI 函数调用、Anthropic 工具使用、MCP——已成为主流，但仅适用于单次调用。模式门控编排将验证扩展至组合工作流，在执行前对多步骤计划施加结构、依赖关系和类型约束。关键区别在于不透明与透明的组合：封装多个内部步骤的预先整理的 MCP 工具提供了已定义接口，却隐藏了内部执行序列；而模式门控工作流会将其展现为经过验证的有向无环图，其结构和步骤间类型均可由机器检查（§5.3）。用户或大语言模型可以替换某个步骤、添加阶段或更改参数，而系统会重新验证整个计划。

5 参考架构

本节介绍一种参考架构，用以实现模式门控编排及其三项操作原则（§4），并提出一种可能将 ED/CF 权衡解耦的设计。

5.1 架构概述

参考架构将职责划分到不同层中，并为各层提供明确定义的接口，使组件能够独立演进（例如，更换大语言模型提供商而不改变执行引擎）。其核心原则是将经模式验证的注册表与对话层分离，并通过编排控制器将两者连接起来；该控制器负责强制执行验证并生成结构化溯源信息。聊天优先界面直接呈现模式门控边界，使验证状态、执行进度和澄清循环一目了然。

5.2 编排控制器

编排控制器在自然语言交互与已验证的执行之间进行协调。随着对话推进，控制器会汇集上下文——聊天历史、先前的工具调用及返回的结果——大语言模型据此在每轮对话中选择恰当的平台操作。只有当用户对拟议操作感到满意且 所有模式约束条件均得到满足后，才会派发执行任务。

两类可调用对象运行在不同层级。平台操作—`search_workflows`,`get_parameters`, `list_datasets`, `execute_workflow`—让大语言模型能够浏览并操作系统。领域工具 是构成工作流的原子科学操作。两者都经过模式验证；平台操作决定用户如何发现并触发工作，领域工具则决定实际执行的内容。

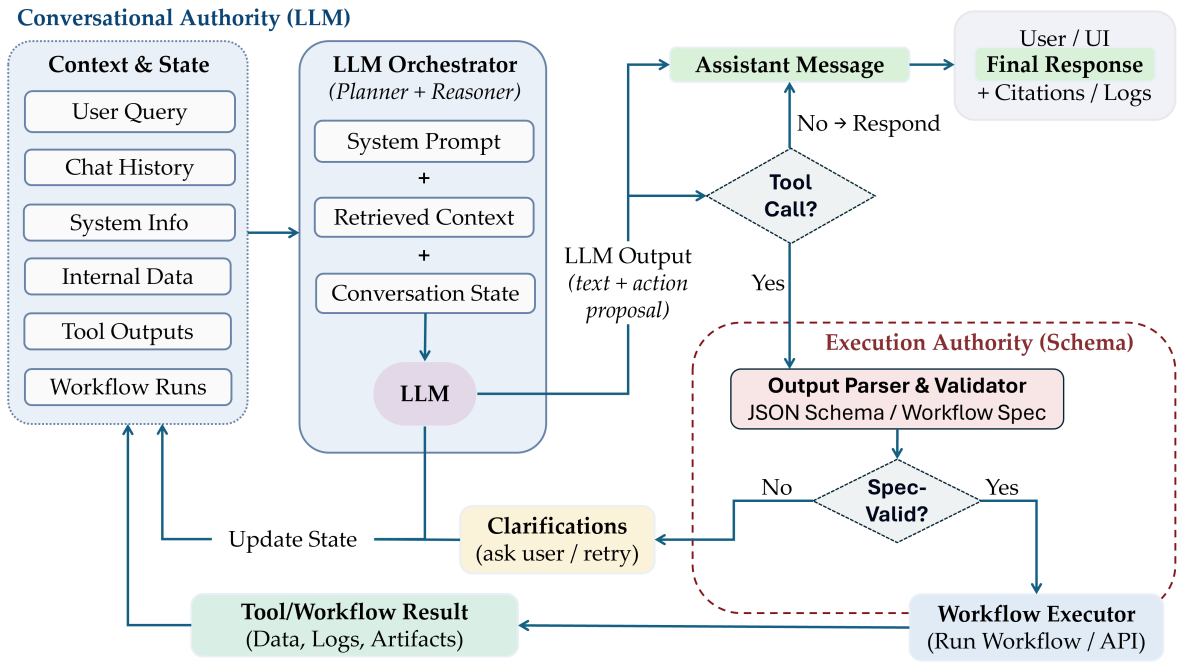

图3：将对话权限与执行权限分离的参考架构。上下文和状态（用户查询、聊天历史、系统信息、内部数据以及先前的工具/工作流输出）被汇总后提供给大语言模型编排器（规划器/推理器），由其生成助手消息，并可选择生成操作提案。提案进入执行权限闸门（红色虚线区域）：输出解析器根据 JSON Schema 或工作流规范对其进行验证。无效提案会触发澄清循环；有效提案则转发给工作流执行器。产生的数据、日志和产物会更新共享上下文，以供后续轮次使用。大语言模型可以自由进行对话，但无法执行操作——执行权限完全由模式验证掌控。

控制器的职责范围是意图理解和执行调度，而非工作流构建。编排器使用来自注册表的已验证工作流；它不会在会话期间创建工作流。以下调用展示了一个用于调度已验证工作流的平台操作：

```
{"tool": "execute_workflow",
"workflow_id": "alloy_inverse_design","parameters": {
"dataset_id": "123e4567-e89b-12d3-a456-426614174000",
"target_properties": ["yield_strength", "creep_life"],
"constraints": {"Cr": {"max": 12.0}, "Co": {"min": 5.0}},
"n candidates": 50}}
```

这些调用对象将模型的影响限制在可审计且符合模式的操作范围内。图 3 展示了由此形成的架构：大语言模型编排器拥有对话权限，而执行权限则由模式验证掌握，因此每个操作提案都必须通过执行权限关卡，才能开始任何计算；每次执行结果也会反馈至上下文，供后续轮次使用。

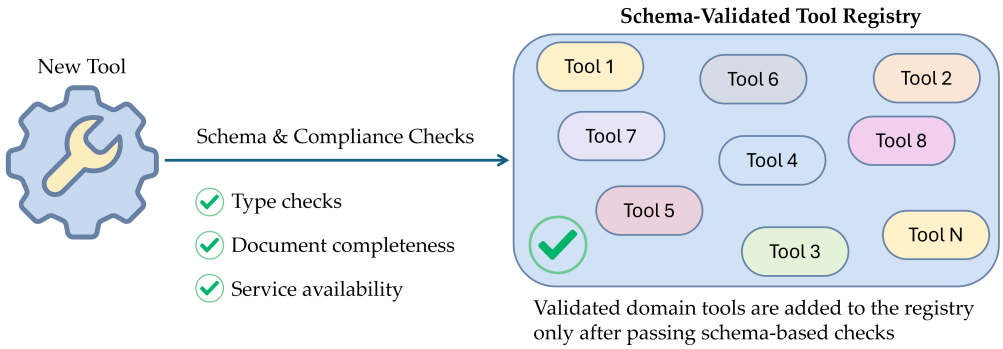

图 4：领域工具模式验证及其纳入注册表。新增领域工具在添加到已验证工具注册表之前，须通过模式和合规性检查，包括类型验证、文档完整性和服务可用性检查。只有经过模式验证的领域工具才能用于工作流执行。

5.3 验证框架

领域工具和工作流以 Python模型的形式声明，并编译为机器可校验模式（例如 JSON Schema），且与其实现一同进行版本化。

由于每个领域工具都声明了类型化输入、输出和依赖约束（§4.1），依赖图便限定了规划搜索空间：对于模式声明了上游要求的目标工具，依赖图会确定所需组件及其顺序，从而将组合问题从组合搜索转变为由结构引导的遍历。因此，组合并不限于手动编写：大语言模型可以提出候选流水线，约束求解器可以枚举有效排序，领域专家也可以手动组装各个步骤；无论采用哪种方式，都可以通过相同的模式级检查验证结果。现有工作流可作为可复用模板，供用户通过替换或重新参数化进行调整。

5.3.1 领域工具模式结构

领域工具定义原子操作，例如数据预处理、模型训练或统计分析。领域工具模式类似于 MCP 工具定义，包含类型化输入、输出和可调用接口，但还带有更丰富的元数据：依赖声明、溯源信息（来源、版本、维护者）、领域标签以及类型化的步骤间契约。这些元数据使工具能够组合成工作流：由于每个工具都会公布其输入/输出类型及领域上下文，编排器便可以匹配兼容的工具，而无需依赖手动连接。在通过参数一致性、文档完整性和服务可用性的自动检查之前，领域工具不能加入注册表（图 4）。完整示例见附录 D.1。

5.3.2 工作流架构结构

工作流将领域工具组合成有向无环图，用以规定执行顺序、依赖项和数据流。工作流架构定义了各个步骤、步骤调用的工具、步骤间依赖项以及参数映射。架构还记录工作流级元数据（例如标签和复杂度评级），以便发现。执行前，引擎会验证图的无环性、步骤间类型兼容性、参数解析情况以及工具是否可用（图 5）。附录 D.2 中提供了一个扩展示例。

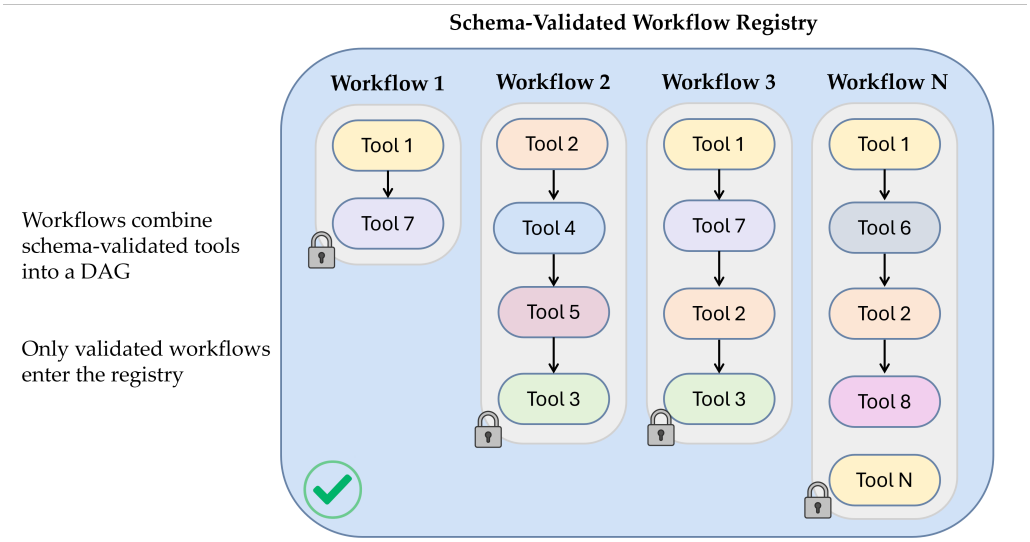

图5：经验证的注册表中的工作流架构组合。每个工作流都将经模式验证的领域工具组合成一个有向无环图，定义步骤之间显式的数据流和参数流。只有通过无环性、类型兼容性和参数解析检查的工作流，才会被纳入经验证的工作流注册表。

5.3.3 模式门控执行

用户选择（或系统推荐）工作流后，控制器会加载相应的架构，以确定必须通过对话收集哪些参数。只有在提供必需输入，并验证其类型正确性、允许值和结构一致性之后，才会开始执行（图6）。

5.4 执行引擎与溯源

执行引擎异步运行已验证工作流，并为每次运行创建溯源记录——记录工作流定义快照、已解析参数、时间戳、环境元数据以及任何故障。由于输入和输出均为模式类型化，记录具有结构化且可查询的特性，从而支持跨运行比较和长期可复现性。

5.5 实践中的模式门控编排

以下交互轨迹完整展示了模式门控模式。

表 5 展示了核心循环。用户提出自然语言请求；系统不会生成代码，而是调用平台操作（`search_workflows`），并将结构化结果返回到对话中。这是第一道边界：即使是“确定要运行什么”，也须通过已验证的平台操作来完成，而不是依赖隐式推理。

选定工作流后， `get_parameters` 系统会检索参数模式，并在对话中补齐必填字段。只有调用完整且类型有效时，才会执行；缺失或类型错误的值会触发澄清（§4.1）。

这一边界也支持快速迭代：编辑是对显式参数的修改，而不是对生成的代码进行更改，因此每次运行都是一次具体的调用，可以直接进行比较。第 9–12 轮对此进行了说明：当用户说“尝试留一法，而不是五折交叉验证”时，系统会复用先前的调用，仅更改一个参数，从而无需代码检查即可进行系统性的消融。

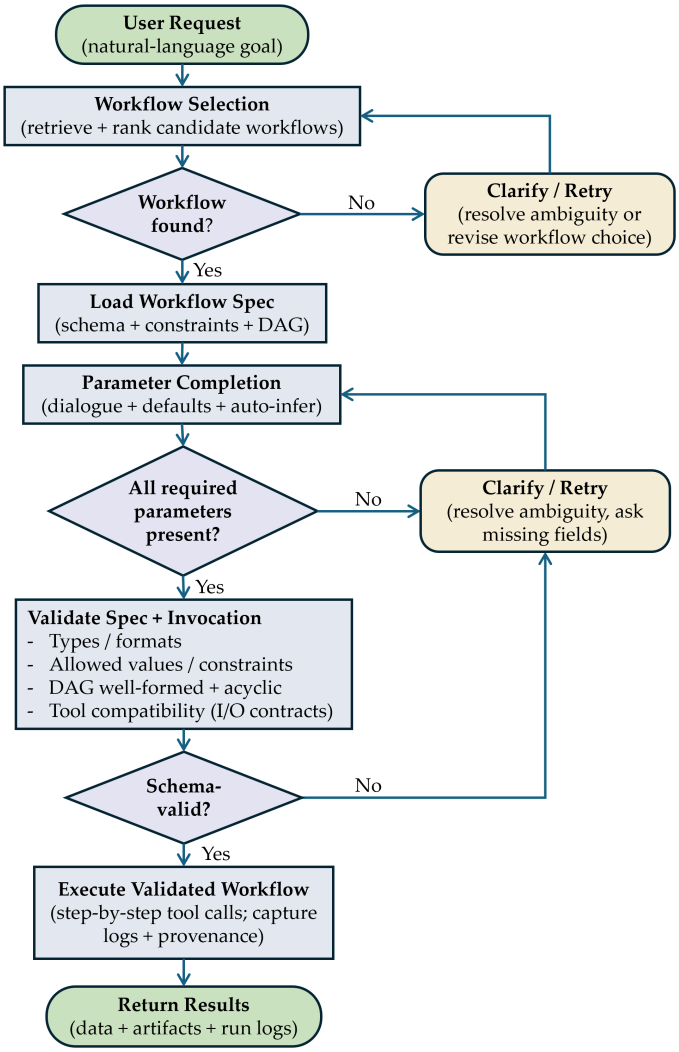

图6：模式门控工作流执行流程。系统识别候选工作流、加载相应模式，并通过对话收集和验证参数；只有满足所有约束条件后才会执行。验证失败会触发澄清循环；对话上下文在整个过程中持续更新。

6 讨论

我们围绕模式门控执行所带来的能力、其成本与局限、治理影响以及开放问题来组织讨论。

6.1 模式门控执行能够实现

为何模式门控能够实现权衡解耦。经验帕累托前沿（图2）的形成源于经审查的系统将对话权限与执行权限耦合在一起：赋予自然语言直接控制运行内容的权力，往往会放弃执行前验证；而强制验证则往往会限制对话。模式门控编排将这两个维度解耦。在规划模式下，模型可以不受约束地进行推理；在行动模式下，未经模式验证，任何内容都不会执行。因此，该架构既继承了以工作流为中心的系统在执行确定性方面的优势（显式、已验证、版本化的产物），又保留了对话优先系统在对话灵活性方面的优势（自然语言驱动发现、参数补全、迭代优化）。其残余约束在于，交互会被引导至注册表能够表示的操作，从而排除不受约束的

表5：实践中的模式门控编排（交互轨迹）。每一行展示代表性对话中的一轮；模式门列标识系统继续执行前强制实施的验证检查点。行颜色用于区分参与者：蓝色 =用户，绿色 =系统。

Turn	执行者	操作/工具调用详情		模式门
1	User	自然语言请求	“我想设计一种低铬含量的新型高温合金……”	—
2	系统	`search_workflows`	注册表查询返回按排名排列的候选工作流	工具模式 验证查询 参数
3	User	选择工作流	从结果中选取“Alloy Design Pipeline v2.1”	—
4	系统	`get_parameters`	检索工作流参数模式（必填字段、类型、范围）	工具模式 验证工作流 ID
5	系统	参数提示	列出必填字段：`model_id`，`target_properties`，`constraints`	—
6	User	提供参数	提供模型标识符、目标和组成约束	—
7	系统	验证	检查类型正确性、允许范围以及结构一致性性	工作流架构 强制执行所有约束条件
8	系统	`execute_workflow`	调度已验证的调用；流式传输进度和结果	阻止直至门7通过
结果反馈至对话上下文				
9	系统	结果摘要	返回指标（R^2、均方根误差）和排名靠前的候选组成加入对话	—
10	User	迭代优化	“试试留一法，而不是五折交叉验证”	—
11	系统	参数更新	复用先前的调用，仅更改一个参数（`validation_strategy`）	工作流架构 重新验证已修改调用
12	系统	`execute_workflow`	分发变体调用；两次运行可直接比较	阻塞，直到第 11 道门控通过

评分标准为对话灵活性保留的生成 = 5 （表 2）——但正是这一约束使执行确定性成为可能。

已部署的模式门控系统能否在实践中实现这种解耦，尤其是在实际工作负载下，对话摩擦、覆盖缺口和注册表维护成本是否会削弱任一维度，仍是一个有待实证研究的问题。我们将部署层面的测量（覆盖缺口频率、参数补全延迟、任务完成率）列为近期最紧迫的优先事项（§6.5）。

作为架构副产品的边界属性。由于前沿反映的是对话权限与执行权限之间的耦合，而非理论上的界限，这种解耦使 §2.2中指出的边界属性——人工审批与溯源——成为调用对象的架构副产品，而非必须单独设计的功能。

6.2 成本与局限

核心代价在于覆盖范围。模式门控系统只能执行其注册表所定义的内容；缺少工具或工作流时，就必须拒绝执行，或转入显式编写路径。这使工作重心从提示工程转向注册表维护——模式设计、验证、版本控制以及向后兼容演进——这是一项组织层面的承诺，在领域边界处尤为重要。成熟的工作流可以将开销分摊给众多用户；而前沿研究

可能恰恰在最需要灵活性的地方缺乏覆盖范围。

三项具体的近期缓解策略可以在不放弃模式门控不变式的前提下，缓和这一硬性边界：(i) 基于模板的引导，即将经验证的工作流作为可组合模板，用户可以通过工具替换或重新参数化来调整这些模板，而无需从头编写；(ii) *LLM* 辅助模式草拟，即由模型根据现有代码、文档或函数签名生成符合模式的 `ToolDefinition` ，并接受与手动编写定义相同的自动化验证关卡（附录 D.1）；以及 (iii)分级注册表，通过区分信任级别，使社区贡献的工具能够在显式的溯源和审查政策下与精选工具共存。至于进一步允许沙箱化生成式执行的混合模式能否保留有意义的保证，仍是一个悬而未决的问题（§6.5）。

从长期来看，最有前景的路径是构建一个联邦式生态系统：如果 MCP 等协议 [58]扩展为支持工具间依赖声明和类型化数据契约，提供方就可以发布符合模式的工具包，由编排器将其组合成经验证的跨提供方工作流。这样一来，覆盖范围的扩展将依赖社区，而不是任何单个团队的注册表（§6.5）。

此外还有三项限制。首先，执行前澄清会引入对话摩擦：每次执行前的结构化协商可能会减缓高参数工作流中的探索。这可以通过依据元数据或先前运行预填充字段，并在各次迭代中沿用先前已确定的参数来缓解（表5，第10–12轮）。其次，模式可强制参数类型正确并保证结构有效性，但无法确保科学适切性——用户对假设、数据质量和模型有效性的判断仍然至关重要。第三，模式门控在边界处提供架构确定性；端到端确定性还需要受控随机种子、容器和固定版本的依赖项。

6.3 治理与安全影响

将所有操作经由可审计的、经模式验证的接口进行路由，可消除一大类治理风险——通过对话进行的任意代码执行——但残余风险仍然存在：危险的已注册工具、针对工具调用的提示注入，以及数据泄漏。

访谈数据凸显了这些关切。安全与知识产权保护是讨论次数排名第四的主题（67次提及，10位利益相关者中有9位；表1），从业者提出了专有配方的保密、出口管制限制，以及对话上下文可能将商业秘密暴露给云端托管的大语言模型的风险。模式门控编排从结构上应对了若干关切：每一项操作都由已验证的调用对象进行中介，从而能够在工具和参数层面实施基于角色的访问控制，借助数据分类标签防止敏感值到达外部服务，并通过防篡改审计跟踪将每次执行映射到版本化模式和经身份验证的身份。

尽管如此，仅靠模式门控是不够的。纵深防御仍然至关重要：最小权限的工具设计、针对敏感操作的审批关卡、防范提示注入的输入净化，以及面向本地部署的网络级隔离。注册表本身的治理——谁可以发布、审查和退役工具——是一个策略设计问题，与开源软件包生态系统面临的挑战相呼应。

6.4 设计指导与开放问题

综述与需求为构建可信的对话式工作流系统提供了实用指导：

- 将已验证的调用 作为执行、溯源和审批的单元。
- 让边界故障清晰可见，并以对话形式提供可执行的处理建议（缺失字段、约束违反、不支持的操作）。

- 将注册表生命周期管理（测试、版本控制、模式演进）作为一流的工程实践进行投入。
- 利用对话降低交互开销（用于发现和参数补全），而非定义可执行行为。

主题数据进一步印证了这些优先事项：系统集成是讨论最多的主题（191 次提及，10/10 位利益相关者），这证实了连接器覆盖范围是必要前提；工作流与任务自动化以及数据搜索与检索的讨论范围也同样广泛。渐进式自主性（19 次，5/10 位）和数据聚合与匿名化（10 次，4/10 位）等覆盖面较窄的主题是值得关注的新兴问题，但在发布初期无需优先考虑。

仍有若干开放问题：注册表应由中央统一维护、支持社区扩展，还是采用分级模式；如何在保持稳定与快速新增能力之间取得平衡；以及当注册表覆盖范围不足时，系统应如何传达不确定性。

6.5 未来工作

内部研究试验台（该架构的一个实例）正在开发中。多项 al 方向 值得调查：
1. 拓展利益相关者招募范围，超越 Intellegens 网络，面向依赖生成式或临时性工具的组织。2. 联邦式工具生态系统：扩展 MCP 等协议，加入工具间依赖声明和类型化数据契约，使软件即服务（SaaS）提供商和研究团队能够发布经过审核、符合模式的工具包，供编排器自动组合成跨提供方工作流——这是扩大注册表覆盖范围、突破单一组织限制的主要方向（§6）。3. 扩大经评估的系统范围，尤其是中等水平的系统（CF = 2–4），以便完善帕累托前沿。4. 对已部署的概念验证系统进行实证评估：模式编写负担、覆盖缺口频率、对话式工作流的使用阻碍，以及用户任务完成时间——这是当前最紧迫的近期优先事项，可用于衡量模式门控系统在 ED/CF 空间中的实际位置。5. 定期纵向重新评估 ED/CF 格局，并记录系统版本，以追踪平台新增功能时帕累托前沿的演变。6. 跨领域交互轨迹：表 5 中的交互轨迹展示了这一模式在材料科学领域的应用；在更多领域加以复制，将有助于强化其领域通用性这一主张。Bioinformatics 是一个自然的首选拓展领域：多步骤流水线（例如，读段质量控制（QC）→ 序列比对 → 变异检出 → 注释）已在 nf-core 等平台中表示为经过验证的有向无环图 [59], ，模式门控模式可为这些流水线提供对话式参数补全和跨步骤类型检查，同时保留现有的溯源基础设施。食品科学和半导体加工也有类似机会，可在不重新实现既有工具链的情况下，为其添加模式封装。7. 借助人工智能辅助工具与工作流编写：如果大语言模型能够按需起草符合规范的模式或组合新颖的工作流有向无环图——并接受相同的验证和审核关卡——那么注册表覆盖范围就会从硬性约束转变为软性约束，有望突破模式门控所造成的对话灵活性上限。8. 混合执行模式：在沙箱中运行的生成式“逃生出口”能否与模式门控的保障机制共存。9. 评分标准验证与自我评估：评估 ED/CF 评分标准能否成为从业者和平台供应商可靠的自我评估工具，帮助他们确定自身系统在设计空间中的位置，并为架构路线图提供参考。

致谢

我们感谢参与访谈并提供反馈的研究团队和研发组织；这些反馈为本研究的需求和设计考量提供了依据。我们也感谢 Intellegens 的同事在本研究开展期间参与讨论并给予支持。

贡献

J.S.（Joel Strickland）构思并主导了本研究，制定了调查框架和访谈流程，开展了访谈，综合整理了需求，设计了多模型大语言模型评分方法和跨模型一致性分析，评估并评分了全部20个软件系统，设计了参考架构，开发了架构原型，开展了主题分析和统计分析，并撰写了论文。A.V.（Arjun Vijeta）协助开展访谈，参与需求综合整理和架构讨论，并提供工程与实施支持。C.M.（Chris Moores）提供工程与实施支持，并就最佳实践提供意见。O.B.（Oliwia Bodek）提供写作支持、编辑协助和概念性反馈。B.N.（Bogdan Nenchev）、T.W.（Thomas Whitehead）、C.P.（Charles Phillips）、K.T.（Karl Tassenberg）和G.C.（Gareth Conduit）审阅了论文，提出意见和改进建议，并提供领域及技术反馈。B.P.（Ben Pellegrini）获得了项目经费，并提供指导与监督。所有作者均审阅并批准了论文终稿。

利益冲突

J.S.、A.V.、B.N.、T.W.、C.P.、K.T.、G.C. 和 B.P. 均隶属于 Intellegens。该公司开发并商业化 AlchemiteTM 平台 [60]。访谈参与者通过 Intellegens 的客户关系招募（§2）；AlchemiteTM 平台并非本研究提出的系统。O.B. 和 C.M. 是独立研究者。作者声明不存在其他竞争利益。

A 访谈方法

A.1 招募、保密与研究背景

参与者通过 Intellegens 的专业网络招募（表 A.1）。访谈广泛探讨了研发领域的工作流程挑战以及对对话式人工智能的期望，而非任何特定商业平台。所有参与者均受保密协议约束；我们仅以汇总形式报告研究结果，不注明受访者身份。在说明研究目的后，我们邀请参与者参加，且他们均自愿参与。

招募范围与局限性。所有利益相关者均通过 Intellegens 现有的客户关系招募，其中数位作者是 Intellegens 的员工（§ 竞争性利益）。这种便利抽样使我们得以接触经验丰富的研发从业者，但也限制了研究结果的可推广性：样本可能过度代表已投资于机器学习辅助工作流的组织，而依赖生成式工具或临时工具的组织则可能代表不足。此处提出的系统并非任何 Intellegens 产品，问题均围绕通用工作流程挑战展开。通过更广泛的招募开展独立重复验证，将有助于提升外部效度（§6.5）。

受访者知道有多位利益相关者参与，但未获知彼此的身份。所有人接受了相同的访谈结构，并回答同一组问题。除专业角色和背景外，未收集任何个人数据。

表 A.1：匿名化的参与者人口统计信息。营收和员工人数为根据公开数据估算的大致区间（2025财年或最新可用数据）。独立顾问的职业经历涵盖多家特种化学品企业。化名随机分配。

ID	行业领域	营收（美元）	员工专家	
组织-1	食品与饮料	>$50B	>100,000	1
组织-2	专业工具	$5–10B	10,000+	2
组织-3	包装与材料	$5–10B		2
组织-4	食品配料	$5–10B	10,000–15,000	6
组织-5	消费品	$5–10B	5,000–10,000	1
Org-6	特种化学品	10–50亿美元	5,000–10,000	2
Org-7	半导体材料	10–50亿美元	5,000–10,000	1
Org-8	建筑化学品	∼$1B	1,000–5,000	1
Org-9	工业印刷	<$1B	1,000–5,000	1
特种化学品顾问（个体）		—	—	1
总计				*18*

A.2 访谈结构

我们为每位利益相关者进行了至多两次访谈：

- **第 1 节：工作流程挑战与自动化机会。**聚焦于当前的计算工作流和摩擦点：耗时环节、手动步骤、集成痛点，以及尝试替代方案的成本过高而无法进行探索的情形。问题涵盖数据准备、模型构建、分析、报告编制和交接。我们

还探讨了自动化（从引导式协助到自主执行）可以在哪些方面加快工作进程。

- **第2场：对研发领域智能体人工智能的期待。** 考察了受访者对对话式人工智能和智能体人工智能的期待：期望的自主程度、信任与控制边界、治理要求、偏好的交互方式（分步式或自主式）、透明度需求、部署限制（云端／混合部署／本地部署），以及与现有系统所需的集成。

访谈按照指定的访谈指南和问题安排进行，同时允许通过追问进行澄清。

A.3 分析方法

主题根据原始访谈记录（而非中间摘要）进行编码。访谈记录被解析为发言轮次段落；剔除访谈者和内部团队的发言后，剩下2,468个外部参与者段落作为编码单元。

A.4 编码规程与编码手册

每位外部参与者的每段发言均构成一个编码单元（共2,468个段落）。在开展系统编码之前，通过反复阅读所有访谈记录，编制了一份包含17个主题的编码手册（表 A.2）。每个主题均由标签、范围描述和关键词模式定义。

编码采用基于正则表达式的模式匹配，并统一应用于所有段落；每个主题包含 20 个模式（参见 §2.1中的理由）。如果某个关联模式匹配成功，则将该段落编码为相应主题；一个段落可以适用多个主题。共计 1,135 次编码按利益相关方广度和提及频率汇总（表1）。

方法学说明。 我们采用的是系统性内容编码，而不是反思性主题分析的迭代式解释过程 [57]。基于正则表达式的匹配确保了可复现性，但引入了精确率—召回率权衡：稀疏主题可能会少计概念上相关的表述，而丰富主题可能会多计偶然提及。将每个主题的模式数量固定为 20 个，是以牺牲精确率为代价来实现主题间的可比性。

为量化这一权衡，我们开展了三部分敏感性分析（完整结果见附录 B）：

1. 留一模式稳定性。 跨主题的平均最大单模式剔除幅度为 32.8%，其中安全与知识产权（64.2%）和渐进式自主性（52.6%）的依赖程度最高
概念范围较窄的主题。每种模式的平均降幅仅为 4.1%，表明冗余度适中。

2. 跨会议收敛效度。 比较两轮访谈（$n_1 = 1$，023, n_2 =1，445 段落）得出 $\rho = 0.487$ ($p = 0.047$)，证实了各场次间主题显著性的一致性。
3. 自助法置信区间。 重抽样（1,000 次迭代）得出较窄的 95% 持续集成（例如，系统集成：191，持续集成 [166, 218]；渐进式自主性：19，持续集成 [11, 28]），证实了频率估计的稳定性。

抽查（每个主题10个段落）得出86–93%的精确率；主要假阳性来源是多义模式（例如，`reason(ing)?` 匹配“有理由相信”）。推动架构类别的主题排名是稳健的；绝对计数应视为指示性估计。

表 A.2：编码手册：用于对从业者访谈回答进行编码的主题定义。行颜色表示主题组：确定性、灵活性、跨领域或范围之外（见表 1）。

主题	描述／编码范围
系统集成	与指定系统的集成：实验室信息管理系统、电子实验记录本、SQL、API 等。
工作流与任务自动化	自动化重复性研发任务：实验设计（DOE），文献综述、报告生成、实验设计
数据清洗与准备	自动化数据清洗、格式化、标准化、质量检查，异常值剔除
安全与知识产权保护	数据安全、知识产权保护、保密、合规、访问控制
领域／科学专业知识	人工智能对领域特定上下文、科学知识、行业标准
安全与伦理边界	绝不应自动化的任务：安全、伦理、战略决策，合规
数据搜索与检索	跨源搜索，历史数据访问，知识发现，检索信息
自然语言界面	对话式人工智能交互、自然语言处理查询、类似聊天机器人的交互
可视化与报告	图表、绘图、PowerPoint/Word 输出、结果的可视化沟通
多智能体／智能体间	多个智能体协同工作、智能体间通信，编排
可解释性与透明度	可解释的 AI 推理，展示 data 来源，论证建议、步骤可见性
人工监督与控制	人在回路、批准/撤销步骤、覆盖人工智能、保持决策权限
渐进式自主性	先保持较高的控制力度，随着信任的建立，再逐步提高自主性
成本与效率	投资回报率、成本优化、资源效率、时间节省、加速 R&D
采用与变革管理	用户采用、培训、文化障碍、怀疑态度、学习曲线
部署模式偏好	本地部署、私有云、混合部署、云托管、桌面部署偏好
数据聚合与匿名化	对敏感客户数据进行聚合与匿名化，以供共享使用

B 主题编码 敏感性分析

本附录报告了在§A.4中概述的完整三部分敏感性分析，其针对相同的2,468个段落以及 17×20 正则表达式编码手册进行。代码和数据可在补充材料中获取。

B.1 模式贡献与留一稳定性

在总计 340 个正则表达式模式（每个主题 20 个）中，191 个（56%）至少匹配了一个段落（每个主题平均有 11.2 个活跃模式）。表 B.1 列出了每个主题的提及总次数、移除单个模式所造成的最大百分比降幅，以及影响最大的模式。

表 B.1：留一模式稳定性。最大降幅是指从该主题的 20 个模式组成的模式集中移除任意单个模式后，编码段落数量出现的最大百分比降幅。

主题	提及次数	最大降幅	平均降幅	影响最大的模式
系统集成	191	23.0%	3.9%	`API[s]?`
工作流自动化	112	29.5%	4.1%	`DOE`
数据搜索与检索	108	17.6%	4.2%	`quer(y|ying|ies)`
自然语言界面	86	23.3%	4.7%	`prompt(s|ing)?`
可解释性	83	30.1%	4.5%	`explain(ability|able)?`
数据清洗	70	32.9%	4.4%	`data\s+prep`
安全与知识产权	67	64.2%	4.1%	`secur(e|ity)`
可视化	65	21.5%	3.9%	`plot(s|ting)?`
领域专业知识	62	51.6%	4.7%	`formulat(ion|e|ing)`
成本与效率	58	19.0%	4.6%	`time\s+(saving|spent)`
采用	51	33.3%	4.1%	`adoption`
人工监督	44	29.5%	3.4%	`human.in.the.loop`
部署偏好	43	23.3%	3.5%	`cloud.?host`
多智能体	41	39.0%	4.8%	`agentic.*orchestrat`
安全与伦理	25	28.0%	4.0%	`never\s+automat`
渐进式自主性	19	52.6%	4.0%	`(build|grow).*trust`
数据匿名化	10	40.0%	4.0%	`aggregat.*data`

三个主题——安全与知识产权（64.2%）、渐进式自主性（52.6%）和领域专业知识（51.6%）——表现出较高的单模式集中度。这些主题在概念上范围较窄，其词汇围绕核心关键词聚集（安全、信任构建、表述）。其余14个主题的最大降幅均低于40%，且在所有情况下，每种模式的平均降幅为 $\leq$5%，表明模式集具有足够的冗余度。

B.2 跨会议聚合效度

两场会议（$n_1 = 1{,}023$，$n_2 = 1{,}445$ 个段落）得出 $\rho = 0.487$ ($p = 0.047$)，证实尽管结构不同且第2场进行了更深入的探查（系统集成的提及次数从 49 次增至 142 次，安全从 12 次增至 55 次），主题显著性仍保持一致。主要出现在某一场会议中的主题（例如，部署偏好：1 次对 42 次；多智能体：2 次对 39 次）反映了第 2 场对智能体人工智能的关注。

B.3 Bootstrap 置信区间

对2,468个段落进行有放回重抽样（1,000次迭代），得到表B.2中的95%置信区间。

表 B .2：主题提及次数的自助法95%置信区间（1,000 次迭代）。

主题	观察值	95% 持续集成
系统集成	191	[166, 218]
工作流自动化	112	[94, 132]
数据搜索与检索	108	[88, 128]
自然语言界面	86	[68, 104]
可解释性	83	[66, 101]
数据清洗	70	[54, 88]
安全与知识产权	67	[51, 84]
可视化	65	[50, 80]
领域专业知识	62	[47, 78]
成本与效率	58	[44, 74]
采用	51	[37, 65]
人工监督	44	[33, 57]
部署偏好	43	[31, 55]
多智能体	41	[29, 54]
安全与伦理	25	[16, 35]
渐进式自主性	19	[11, 28]
数据匿名化	10	[5, 17]

置信区间半宽中位数为观测值的 ≈15%。前五名排名在全部1,000个样本中均得以保持，证实架构结论并非特定段落或公司所造成的假象。

C 评分协议

以下提示词已提交给 ChatGPT 5.2（OpenAI，2025年）、Claude Sonnet 4.6（Anthropic，2025年；通过 Perplexity）和 Gemini 3.1 Pro（Google，2025年；通过 Perplexity）——均启用思考模式——每个系统在无先前评分历史的全新上下文中各提交一次（每个模型五次会话，共计 15 次）。评分标准与表2相同。

系统规则
你正在使用下方评分标准，从两个架构维度对一个软件系统进行评分。请依据该系统公开文档所载的默认内置能力进行评分。不要基于假设性配置或自定义扩展进行评分。请使用网络搜索，借助官方文档、产品页面、学术论文或可靠的技术来源核实相关能力。优先采用一手资料。如不确定，请选择较低分数并说明你的推理。每一项分数都必须以客观、可观察的证据，或用于推断的清晰推理作为依据。

待评分系统
名称： [系统名称]网站/文档： [
URL(s)]版本/层级： [以公开文档为准]

评分标准
请按1–5的有序量表对每个维度进行评分：

执行确定性（*ED*）：
1 = 不受约束的生成代码；没有正式的执行前门控2 = 主要依赖运行时检查；执行前约束较弱3 = 开发者定义的工具接口；部分验证，但并未统一受模式约束4 = 经模式/类型验证的工具调用，并进行强有力的执行前验证5 = 显式工作流规范/注册表，配合静态验证；通过构造实现可复现性

对话灵活性（*CF*）：1 = 没有对话式界面（仅支持领域特定语言/配置/图形用户界面）2 = 仅提供有限的自然语言辅助（表单帮助、搜索、提示）；自然语言不控制执行3 = 自然语言可辅助编写/修改工作流；执行需要外部化制品4 = 自然语言可通过结构化调用直接选择/设置工具/工作流调用的参数5 = 自由形式自然语言或智能体循环可直接驱动操作，且结构约束极少

所需输出
对于每个维度，请提供：(1) 分数（1–5）；(2) 置信度（高/中/低）；(3) 用 1–2 句话说明理由，并引用具体证据；(4) 如适用，提供边界情况说明。

C.1 逐会话评分

表格 C.1–C.3 按模型家族分组，报告 §3.1中所述 15 场独立评分会话各自的 ED/CF 分数。每场会话都在全新的对话上下文中使用上述提示模板。行颜色表示范式组（见表3）。

表 C.1：逐会话 ED/CF 分数：ChatGPT 5.2（运行 1–5）。每个单元格显示 (ED, CF)。中位数 = 表示模型内中位数。行颜色表示表3中定义的范式组。

标识符	系统	运行 1	运行 2	运行 3	运行 4	运行 5	中位数 执行确定性	中位数 对话灵活性
1	大语言模型聊天（通用）	(1, 3)	(1, 3)	(1, 3)	(1, 3)	(1, 3)	1	3
2	GitHub Copilot	(2, 5)	(3, 5)	(3, 5)	(2, 5)	(2, 5)	2	5
3	AutoGPT 风格智能体	(2, 5)	(2, 5)	(2, 5)	(3, 5)	(2, 5)	2	5
4	ReAct 风格工作流	(3, 5)	(3, 5)	(3, 5)	(3, 5)	(3, 5)	3	5
5	LangChain（典型）	(3, 5)	(3, 5)	(3, 5)	(3, 5)	(3, 5)	3	5
6	SciAgents	(3, 5)	(3, 5)	(3, 5)	(2, 5)	(3, 5)	3	5
7	语义内核	(3, 4)	(3, 4)	(3, 4)	(3, 4)	(3, 4)	3	4
8	OpenAI 助手	(4, 4)	(4, 4)	(4, 4)	(4, 4)	(4, 4)	4	4
9	Copilot Studio	(4, 4)	(4, 4)	(5, 4)	(4, 4)	(4, 4)	4	4
10	n8n	(3, 3)	(3, 3)	(3, 3)	(3, 3)	(3, 3)	3	3
11	Snakemaker风格	(3, 3)	(3, 3)	(4, 3)	(3, 3)	(3, 3)	3	3
12	Dataiku DSS	(5, 3)	(3, 3)	(4, 3)	(4, 3)	(4, 3)	4	3
13	Galaxy	(5, 1)	(5, 1)	(5, 2)	(5, 1)	(5, 2)	5	1
14	Snakemake	(5, 1)	(3, 1)	(5, 1)	(3, 1)	(5, 1)	5	1
15	Nextflow	(3, 1)	(3, 1)	(3, 1)	(3, 1)	(3, 1)	3	1
16	nf-core	(4, 1)	(4, 1)	(4, 1)	(4, 1)	(4, 1)	4	1
17	WorkflowHub 风格	(4, 1)	(4, 1)	(4, 1)	(4, 1)	(4, 1)	4	1
18	AWS Step Functions	(5, 2)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
19	Apache Airflow	(3, 1)	(3, 1)	(3, 1)	(3, 1)	(3, 1)	3	1
20	Kubeflow / Argo	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1

表 C.2：每场会话 ED/CF 分数：Claude Sonnet 4.6（通过 Perplexity）（运行 1–5）。每个单元格显示 (ED, CF)。中位数 = 模型内中位数。行颜色表示表 3 中定义的范式组。

标识符	系统	运行 1	运行 2	运行 3	运行 4	运行 5	ED 中位数	CF 中位数
1	大语言模型聊天（通用）	(1, 3)	(1, 3)	(1, 3)	(1, 3)	(1, 3)	1	3
2	GitHub Copilot	(3, 5)	(2, 5)	(2, 5)	(3, 5)	(3, 5)	3	5
3	AutoGPT 风格智能体	(2, 5)	(2, 5)	(2, 5)	(2, 5)	(2, 5)	2	5
4	ReAct 风格工作流	(2, 5)	(2, 5)	(2, 5)	(2, 5)	(2, 5)	2	5
5	LangChain（典型）	(3, 5)	(3, 5)	(3, 5)	(3, 5)	(3, 5)	3	5
6	SciAgents	(3, 5)	(2, 5)	(2, 5)	(3, 5)	(2, 5)	2	5
7	语义内核	(4, 4)	(3, 4)	(3, 4)	(3, 4)	(3, 4)	3	4
8	OpenAI 助手	(4, 4)	(4, 4)	(4, 4)	(4, 4)	(4, 4)	4	4
9	Copilot Studio	(4, 4)	(5, 4)	(4, 4)	(4, 3)	(5, 4)	4	4
10	n8n	(4, 3)	(4, 3)	(4, 3)	(4, 3)	(4, 3)	4	3
11	Snakemaker风格	(5, 3)	(5, 3)	(5, 3)	(5, 3)	(5, 3)	5	3
12	Dataiku DSS	(4, 3)	(4, 3)	(4, 3)	(4, 3)	(4, 3)	4	3
13	Galaxy	(5, 2)	(5, 2)	(5, 2)	(5, 2)	(5, 2)	5	2
14	Snakemake	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
15	Nextflow	(5, 1)	(4, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
16	nf-core	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
17	WorkflowHub 风格	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
18	AWS Step Functions	(5, 1)	(5, 2)	(5, 1)	(5, 1)	(5, 1)	5	1
19	Apache Airflow	(5, 1)	(5, 1)	(4, 1)	(5, 1)	(5, 1)	5	1
20	Kubeflow / Argo	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1

表 C.3：每场会话 ED/CF 分数：Gemini 3.1 Pro（通过 Perplexity）（运行 1–5）。每个单元格显示 (ED, CF)。中位数 = 模型内中位数。行颜色表示表中定义的范式组3。

标识符	系统	运行 1	运行 2	运行 3	运行 4	运行 5	中位数 ED	中位数 对话灵活性
1	大语言模型聊天（通用）	(1, 3)	(1, 3)	(1, 3)	(1, 5)	(1, 3)	1	3
2	GitHub Copilot	(1, 5)	(1, 5)	(3, 5)	(1, 5)	(2, 5)	1	5
3	AutoGPT 风格智能体	(1, 5)	(2, 5)	(1, 5)	(2, 5)	(2, 5)	2	5
4	ReAct 风格工作流	(2, 5)	(2, 5)	(2, 5)	(2, 5)	(2, 5)	2	5
5	LangChain（典型）	(4, 5)	(4, 5)	(4, 5)	(4, 5)	(3, 5)	4	5
6	SciAgents	(3, 5)	(1, 5)	(3, 5)	(2, 5)	(3, 5)	3	5
7	语义内核	(3, 4)	(3, 4)	(3, 5)	(3, 4)	(3, 4)	3	4
8	OpenAI 助手	(4, 4)	(4, 4)	(4, 4)	(4, 4)	(4, 4)	4	4
9	Copilot Studio	(5, 4)	(5, 4)	(4, 4)	(4, 4)	(5, 4)	5	4
10	n8n	(4, 3)	(4, 3)	(4, 3)	(5, 3)	(3, 3)	4	3
11	Snakemaker风格	(5, 3)	(5, 3)	(5, 3)	(5, 3)	(5, 3)	5	3
12	Dataiku DSS	(5, 3)	(4, 3)	(5, 3)	(5, 3)	(5, 3)	5	3
13	Galaxy	(5, 2)	(5, 2)	(5, 1)	(5, 3)	(5, 2)	5	2
14	Snakemake	(5, 1)	(5, 1)	(5, 1)	(3, 1)	(5, 1)	5	1
15	Nextflow	(3, 1)	(3, 1)	(4, 1)	(3, 1)	(3, 1)	3	1
16	nf-core	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
17	WorkflowHub 风格	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
18	AWS Step Functions	(5, 3)	(5, 1)	(5, 1)	(5, 1)	(5, 1)	5	1
19	Apache Airflow	(5, 1)	(5, 1)	(5, 1)	(4, 1)	(5, 1)	5	1
20	Kubeflow / Argo	(5, 1)	(5, 1)	(5, 1)	(5, 1)	(4, 1)	5	1

模型内一致性。 ChatGPT 5.2：五次运行中 31/40 项评分完全相同，38/40 项相差不超过一个序数点；Krippendorff's $\alpha_{\text{ED}} = 0.875$, $\alpha_{\text{CF}} = 0.991$。Claude Sonnet 4.6：32/40 项完全相同，40/40 项相差不超过一个点；$\alpha_{\text{ED}} = 0.954$, $\alpha_{\text{CF}} = 0.992$。Gemini 3.1 Pro：25/40 项完全相同，33/40 项相差不超过一个点；$\alpha_{\text{ED}} = 0.885$, $\alpha_{\text{CF}} = 0.957$。

跨模型一致性。 在全部 15 次运行中，19/40 项评分完全相同，28/40 项最多相差一个序数点。Krippendorff's α（序数，将每次运行视为独立评分者）：$\alpha_{\text{ED}} = 0.797$, $\alpha_{\text{CF}} = 0.980$。成对跨平台一致性：ChatGPT +Claude $\alpha_{\text{ED}} = 0.788$, $\alpha_{\text{CF}} = 0.990$；ChatGPT + Gemini $\alpha_{\text{ED}} = 0.777$, $\alpha_{\text{CF}} = 0.973$；Claude + Gemini $\alpha_{\text{ED}} = 0.885$, $\alpha_{\text{CF}} = 0.976$。七个系统的共识中位数相对于仅 ChatGPT 的中位数偏移了一个点；总体帕累托前沿结构和范式聚类保持不变。

反身性说明。 ChatGPT 评分模型由负责其中一个受评系统（ID 8）的同一机构（OpenAI）开发，因此可能存在反身性问题。来自另外两个模型系列（Claude、Gemini）的独立确认直接缓解了这一问题：ID 8 获得了三个模型完全一致的 ED/CF 分数，这与其公开记录的架构（强模式验证、部分溯源）一致。

一个 par 同样的担忧也适用于 Claude Sonnet 4.6（Anthropic），其工具调用 API 和 MCP [5 8]

是模式门控讨论中的特性。没有单独对任何 Anthropic 产品进行评分，因此该风险是间接的。Claude 的评分未显示系统性的执行确定性偏差（成对 α_{ED}：Claude + Gemini $= 0.885$、Claude + ChatGPT $= 0.788$，两者均达成实质性一致），排除 Claude 会话后，帕累托前沿结构保持不变。

所有共识中位数分数均由作者独立确认。

D 模式示例

本附录提供第 5.3 节所述的完整工具和工作流模式。

D.1 领域工具模式示例：材料-性能预测器

以下定义展示了一个领域工具模式。MCP 工具定义将包含 `parameters` 块；其下方的字段——`input_schema`、`output_schema`、`dependencies`、`domain_tags`、`provenance`以及操作元数据——则是实现组合、发现和溯源的附加内容。

```
ToolDefinition(id="materials_property_predictor",
name="Materials-Property Predictor",
description="Train and evaluate surrogate models for
composition-property relationships",version="2.1.0",parameters=[
ParameterDefinition(name="dataset_id",type="str",
description="UUID of uploaded dataset",required=True,
examples=["550e8400-e29b-41d4-a716-446655440000"]),
ParameterDefinition(name="target_properties",type="list[str]",
description="Column names to predict",required=True,
examples=[["yield_strength", "elongation"]]),ParameterDefinition(
name="validation_strategy",type="str",
description="Cross-validation approach",required=False,
default="5-fold",
allowed_values=["5-fold", "10-fold", "leave-one-out"]),],
# --- Fields beyond a standard MCP tool definition ---
input_schema={
"dataset": {"type": "dataframe", "columns": "dynamic"},
"target_columns": {"type": "list[string]"},
```

```
    output_schema={"model_id": {"type": "string"},
"metrics": {"type": "dict", "keys": ["r2_score", "rmse"]},
"predictions": {"type": "dataframe"}},dependencies=["data_loader"],
domain_tags=["materials", "surrogate-model", "regression"],
provenance={"origin": "intellegens-core",
"maintainer": "ml-team@example.org"},estimated_duration=5.0,
   # minutesrequires_network=True)
```

由于模式采用结构化格式，大语言模型可以根据现有文档或代码起草符合要求的 `ToolDefinition`，从而减轻编写负担。只有通过与手动定义相同的自动化验证后，草案才会进入注册表。

D.2 工作流模式示例：基本数据分析

以下 JSON Schema 定义了一个包含三个步骤的加载–清理–分析工作流。每个步骤都会调用一个已验证的领域工具；数据通过参数映射和图边显式流转。工作流级别的 `parameters` 定义面向用户的输入及执行前验证规则；`metadata`支持发现。

```
{"workflow_id": "basic_data_analysis",
"name": "Basic Data Analysis", "description": "Load,  clean,
 and statistically profile atabular dataset", "steps": [{
"step_id": "load_data", "tool_id": "data_loader",
"name": "Load Data",
"description": "Load data from various sources",
"parameters": {"file_type": "csv"}, "dependencies": [],
"estimated duration": 2.0_}, {"step_id": "clean_data",
"tool_id": "data_cleaner", "name": "Clean Data",
"description": "Clean and preprocess the data",
"parameters": {
"operations": ["remove_duplicates", "handle_missing"],
"missing_strategy": "remove"}, "dependencies": ["load_data"],
"estimated duration": 3.0_
```

```
    }, {"step_id": "analyze_data",
"tool_id": "data_analyzer", "name": "Analyze Data",
"description": "Perform statistical analysis",
"parameters": {"analysis_type": "dataset_profile",
"generate_summary": true},
"dependencies": ["clean_data"],
"estimated duration": 5.0}],"parameter_mappings": [{
"from_step": "load_data", "from_parameter": "data",
"to_step": "clean_data", "to_parameter": "data",
"description": "Pass loaded data to cleaning step"}, {
"from_step": "clean_data",
"from_parameter": "cleaned_data",
"to_step": "analyze_data", "to_parameter": "data",
"description": "Pass cleaned data to analysis step"}],
"edges": [{"edge_id": "load_to_clean",
"source_node_id": "load_data",
"target_node_id": "clean_data", "source_output": "data",
"target_input": "data"}, {"edge_id": "clean_to_analyze",
"source_node_id": "clean_data",
"target_node_id": "analyze_data",
"source_output": "cleaned_data", "target_input": "data"
}],"parameters": {"dataset_file": {"type": "string",
"required": true,
"description": "Path to the dataset file",
```

```
    "validation_rules": {"not_empty": true}},
"missing_strategy": {"type": "string","required": false,
"default": "remove",
"description": "Strategy for handling missing data",
"validation_rules": {
"allowed_values": ["remove", "fill_mean", "fill_median"]}},
"metadata": {"complexity": "simple",
"estimated_duration_minutes": 10,
"tags": ["data", "analysis", "basic"],
"categories": ["data_analysis"],
"use_cases": ["Basic data exploration","Quick data insights"]}}
```

这些示例共同展示了这些模式与扁平化工具调用接口的区别：类型化契约支持自动建立步骤间连接；依赖声明使执行顺序可由机器检查；验证规则可强制执行前约束；而溯源元数据则支持发现、审计和可复现性。

参考文献

[1] J. Strickland, B. Nenchev, K. Tassenberg, S. Perry, G. Sheppard, H. Dong, R. Zhang, G. Burca 和 N. D'Souza. 定向凝固镍基高温合金中马赛克性的起源。材料学报，217:117180，2021。doi: 10.1016/j.actamat.2021.117180.[2] J. Strickland, B. Nenchev, S. Perry, K. Tassenberg, S. Gill, C. Panwisawas, H. Dong, N. D'Souza 和 S. Irwin. 单晶合金凝固组织中六方性的本质：机制与应用。材料学报，200:417–431，2020。doi: 10.1016/j.actamat.2020.09.019.[3] J. Strickland, M. Ghisoni, H. Marshall, T. Whitehead, B. Nenchev, B. Pellegrini, C. Phillips, K. Tassenberg, G. Conduit, S. Davey, S. Dorman, D. Ferguson 和 J. Sol. 不确定性程度：在极端环境中利用保形深度学习进行非侵入式核心体温预测。《工程通讯》，4:219，2025年。doi: 10.1038/s44172-025-00548-6.[4] S. Cohen-Boulakia, K. Belhajjame, O. Collin, J. Chopard, C. Froidevaux, A. Gaignard, K. Hinsen, P. Larmande, Y. Le Bras, F. Lemoine 和 F. Mareuil. 生命科学中实现计算可复现性的科学工作流：现状、挑战与机遇。《未来一代计算机系统》，75:284–298，2017。doi: 10.1016/j.future.2017.01.012.[5] T. Hope, D. Downey, D. S. Weld, O. Etzioni 和 E. Horvitz. 科学发现的计算转折点。《美国计算机协会通讯》，66(8):62–73，2023。doi: 10.1145/3576896.[6] Fan Liu, Jindong Han, Tengfei Lyu, Weijia Zhang, Zhe-Rui Yang, Lu Dai, Cancheng Liu 和 Hao Liu. 科学发现的基础模型：从范式增强到范式转变。arXiv预印本 arXiv:2510.15280，2025年。预印本。

[7] S. P. Huber, E. Bosoni, M. Bercx, J. Bröder, A. Degomme, V. Dikan, K. Eimre, E. Flage-Larsen, A. Garcia, L. Genovese 和 D. Gresch. 使用不同量子引擎计算材料属性的通用工作流。*npj Computational Materials*，7(1):136，2021。doi: 10.1038/s41524-021-00594-6.[8] A. M. Ganose, H. Sahasrabuddhe, M. Asta, K. Beck, T. Biswas, A. Bonkowski, J. Bustamante, X. Chen, Y. Chiang, D. C. Chrzan 和 J. Clary. Atomate2：用于材料科学的模块化工作流。《数字发现》，4(7):1944–1973，2025年。doi: 10.1039/D5DD00019J.[9] D. C. Elton, Z. Boukouvalas, M. S. Butrico, M. D. Fuge 和 P. W. Chung. 应用机器学习技术预测含能材料的属性。《科学报告》，8(1):9059，2018。doi: 10.1038/s41598-018-27344-x.[10] R. A. Patel, C. H. Borca 和 M. A. Webb. 利用机器学习进行聚合物序列或组成设计的特征化策略。《分子系统设计与工程》，7(6): 661–676，2022。doi: 10.1039/D1ME00160D.[11] A. L. Lamprecht, M. Palmblad, J. Ison, V. Schwämmle, M. S. Al Manir, I. Altintas, C. J. Baker, A. B. H. Amor, S. Capella-Gutierrez, P. Charonyktakis 和 M. R. Crusoe. 生命科学中工作流自动化组合的展望。*F1000Research*, 10:897，2021。doi: 10.12688/f1000research.54159.1.[12] A. Singh, S. Purawat, A. Rao 和 I. Altintas. 利用机器学习对科学工作流进行模块化性能预测。《未来一代计算机系统》，114:1–14，2021。doi: 10.1016/j.future.2020.04.048.[13] O. J. R. Gustafsson, S. R. Wilkinson, F. Bacall, S. Soiland-Reyes, S. Leo, L. Pireddu, S. Owen, N. Juty, J. M. Fernández, T. Brown 和 H. Ménager. Workflowhub：计算工作流注册表。《科学数据》，12(1):837，2025年。doi: 10.1038/s41597-025-04786-3.[14] J. C. J. Strickland, P. F. D. Woolston 和 T. M. Whitehead. 自适应实验设计。载于《产品配方的数字化转型》。CRC出版社，2024。doi: 10.1201/9781003385974-15.[15] S. Farshidi, X. Liao, N. Li, D. Goldfarb, B. Magagna, M. Stocker, K. Jeffery, P. Thijsse, C. Pichot, A. Petzold 和 Z. Zhao. 异构研究基础设施之间的知识共享与发现。*Open Research Europe*，1:68，2023。doi: 10.12688/openreseurope. 13677.3.[16] L. C. McInnes, D. Arnold, P. Balaprakash, M. Bernhardt, B. Cerny, A. Dubey, R. Giles, D. W. Hood, M. A. Leung, V. Lopez-Marrero 和 P. Messina. 2025年科学计算下一代生态系统研讨会报告：利用社区、软件和人工智能开展跨学科团队科学。arXiv预印本 arXiv:2510.03413，2025年。arXiv.[17] S. Oliveira 和 D. E. Stewart. 《编写科学软件：优良风格指南》。剑桥大学出版社，2006。ISBN 9780521675956.[18] K. Alam, B. Roy, C. K. Roy 和 K. Mittal. 对科学工作流系统开发挑战的实证调查。《实证软件工程》，30(5):151，2025年。doi: 10.1007/s10664-025-10705-2.[19] B. D. Lee. 记录科学软件的十条简单规则。*PLOS*计算生物学，14(12):e1006561，2018。doi: 10.1371/journal.pcbi.1006561。

[20] J. Wang、G. Xiao、S. Zhang、H. Lei、Y. Liu 和 Y. Sui。《深度学习系统中的兼容性问题：挑战与机遇》。载于第31届 ACM 欧洲软件工程会议与软件工程基础研讨会联合会议论文集（ESEC/FSE '23），第476–488页，2023年11月。doi：10.1145/3611643.3616321。[21] A. D. Pimentel、T. Stefanov、H. Nikolov、M. Thompson、S. Polstra 和 E. F. Deprettere。《系统级设计流程的工具集成与互操作性挑战：案例研究》。载于《嵌入式计算机系统：架构、建模与仿真》（SAMOS 2008），《计算机科学讲义》第5114卷，第167–176页。Springer，2008年7月。doi：10.1007/978-3-540-70550-5_19。[22] G. K. Sandve、A. Nekrutenko、J. Taylor 和 E. Hovig。《可复现计算研究的十条简单规则》。PLOS计算生物学，9(10)：e1003285，2013年。doi：10.1371/journal.pcbi.1003285。[23] V. Stodden 和 S. Miguez。《计算科学最佳实践：支持可复现且可扩展研究的软件基础设施与环境》。《开放研究软件期刊》，2(1)：e21，2014年。doi：10.5334/jors.ay。[24] R. C. Jiménez、M. Kuzak、M. Alhamdoosh、M. Barker、B. Batut、M. Borg、S. Capella-Gutierrez、N. C. Hong、M. Cook、M. Corpas 和 M. Flannery。《鼓励研究软件最佳实践的四条简单建议》。F1000Research，6：ELIXIR–876，2017年。doi：10.12688/f1000research.11407.1。[25] J. Freire、D. Koop、E. Santos 和 C. T. Silva。《计算任务的溯源：综述》。《科学与工程计算》，10(3)：11–21，2008年。doi：10.1109/MCSE.2008.79。[26] P. Missier、S. Dey、K. Belhajjame、V. Cuevas-Vicenttín 和 B. Ludäscher。《D-PROV：扩展 PROV 溯源模型以纳入工作流结构》。载于第5届 USENIX溯源理论与实践研讨会（TaPP 13），2013年。URL https://www.usenix.org/conference/tapp13/technical-sessions/presentation/missier。[27] L. Moreau、B. Clifford、J. Freire、J. Futrelle、Y. Gil、P. Groth、N. Kwasnikowska、S. Miles、P. Missier、J. Myers 和 B. Plale。《开放溯源模型核心规范（v1.1）》。《未来一代计算机系统》，27(6)：743–756，2011年。doi：10.1016/j.future.2010.07.005。[28] C. Boettiger。《面向可复现研究的 Docker 入门》。《ACM SIGOPS操作系统评论》，49(1)：71–79，2015年。doi：10.1145/2723872.2723882。[29] B. Grüning、J. Chilton、J. Köster、R. Dale、N. Soranzo、M. Van Den Beek、J. Goecks、R. Backofen、A. Nekrutenko 和 J. Taylor。《生命科学中的实用计算可复现性》。《细胞系统》，6(6)：631–635，2018年。doi：10.1016/j.cels.2018.03.014。[30] J. E. Hannay、C. MacLeod、J. Singer、H. P. Langtangen、D. Pfahl 和 G. Wilson。《科学家如何开发和使用科学软件？》。载于2009年ICSE计算科学与工程软件工程研讨会，第1–8页。IEEE，2009年5月。doi：10.1109/SECSE.2009.5069155。[31] S. Hettrick、M. Antonioletti、L. Carr、N. Chue Hong、S. Crouch、D. C. De Roure、I. Emsley、C. Goble、A. Hay、D. Inupakutika 和 M. Jackson。《英国科研软件调查2014》。Zenodo 数据集，2014年。数据集。[32] M. S. Krafczyk、A. Shi、A. Bhaskar、D. Marinov 和 V. Stodden。《从计算结果复现中汲取经验：提出三项原则并介绍复现软件包》。《英国皇家学会哲学汇刊A辑》，379(2197)：20200069，2021年。doi：10.1098/rsta.2020.0069。

[33] M. Ziemann、P. Poulain 和 A. Bora。《计算可复现性的五大支柱：生物信息学及其他领域》。Briefings in Bioinformatics，24(6)：bbad375，2023年。doi：10.1093/bib/bbad375。[34] R. de Oliveira Andrade。《大规模可复现性项目未能验证数十项生物医学研究》。《自然》，641(8062)：293–294，2025年。doi：10.1038/d41586-025-01266-x。[35] O. Yildiz 和 T. Peterka。《大语言模型会说科学工作流吗？》arXiv 预印本 arXiv:2412.10606，2024年。arXiv。[36] Khairul Alam 和 Banani Roy。《从提示词到流程线：用于生物信息学科学工作流开发的大语言模型》。arXiv 预印本 arXiv:2507.20122，2025年。arXiv。[37] X. Jiang、W. Wang、S. Tian、H. Wang、T. Lookman 和 Y. Su。《自然语言处理和大语言模型在材料发现中的应用》。npj Computational Materials，11(1)：79，2025年。doi：10.1038/s41524-025-01554-0。[38] S. Yao、J. Zhao、D. Yu、N. Du、I. Shafran、K. Narasimhan 和 Y. Cao。《ReAct：在语言模型中协同推理与行动》。arXiv 预印本 arXiv:2210.03629，2022年。[39] Xingyao Wang、Yangyi Chen、Lifan Yuan、Yizhe Zhang、Yunzhu Li、Hao Peng 和 Heng Ji。《可执行代码操作可激发更优秀的大语言模型智能体》。载于第41届国际机器学习大会论文集，《机器学习研究论文集》第235卷，第50208–50232页，2024年。URL https://proceedings.mlr.press/v235/wang24h.html。[40] A. M. Bran、S. Cox、O. Schilter、C. Baldassari、A. D. White 和 P. Schwaller。《为大语言模型配备化学工具》。Nature Machine Intelligence，6：525–535，2024年。doi：10.1038/s42256-024-00832-8。[41] W. Shin、R. Souza、D. Rosendo、F. Suter、F. Wang、P. Balaprakash 和 R. F. da Silva。《智能体人工智能时代科学工作流的（革）新：迈向自主科学》。arXiv 预印本 arXiv:2509.09915，2025年。[42] M. Morishige 和 R. Koshihara。《确保生成式人工智能系统在通用场景中的可复现性：回归测试与开放数据集框架》。arXiv 预印本 arXiv:2505.02854，2025年。[43] M. Hosseini、S. P. Horbach、K. Holmes 和 T. Ross-Hellauer。《生成式人工智能转折点上的开放科学：挑战与机遇的探索性分析》。Quantitative Science Studies，6：22–45，2025年。doi：10.1162/qss_a_00337。[44] S. Baltes、F. Angermeir、C. Arora、M. M. Barón、C. Chen、L. Böhme、F. Calefato、N. Ernst、D. Falessi、B. Fitzgerald 和 D. Fucci。《涉及大语言模型的软件工程实证研究评估指南》。arXiv 预印本 arXiv:2508.15503，2025年。arXiv。[45] M. Brucks 和 O. Toubia。《提示架构会在大语言模型中引入方法学伪影》。PLOS ONE，20(4)：e0319159，2025年。doi：10.1371/journal.pone.0319159。[46] C. Kervadec、F. Franzon 和 M. Baroni。《非自然语言处理：语言模型如何处理机器生成的提示？》arXiv 预印本 arXiv:2310.15829，2023年。[47] Z. Ji、N. Lee、R. Frieske、T. Yu、D. Su、Y. Xu、E. Ishii、Y. J. Bang、A. Madotto 和 P. Fung。《自然语言生成中的幻觉综述》。ACM Computing Surveys，55(12)：1–38，2023年。doi：10.1145/3571730。

[48] F. Jia、T. Wu、X. Qin 和 A. Squicciarini。任务护盾：通过强制任务对齐，防御大语言模型智能体中的间接提示注入。载于第63届计算语言学协会年会论文集（第1卷：长篇论文），第29680–29697页，2025年7月。URL https://aclanthology.org/2025.acl-long.1435/。[49] K. Hines、G. Lopez、M. Hall、F. Zarfati、Y. Zunger 和 E. Kiciman。利用聚焦防御间接提示注入攻击。arXiv预印本 arXiv:2403.14720，2024年。[50] Y. Hu、S. Wang、T. Nie、Y. Zhao 和 H. Wang。理解大语言模型供应链：结构、领域与漏洞。arXiv预印本 arXiv:2504.20763，2025年。[51] B. C. Das、M. H. Amini 和 Y. Wu。大语言模型的安全与隐私挑战：综述。《ACM Computing Surveys》，57(6):1–39，2025年。doi: 10.1145/3712001。[52] J. Goecks、A. Nekrutenko、J. Taylor 和 Galaxy Team。Galaxy：支持生命科学领域中易于使用、可复现且透明的计算研究的综合方案。《Genome Biology》，11(8):R86，2010年。doi: 10.1186/gb-2010-11-8-r86。[53] J. Köster 和 S. Rahmann。Snakemake——一种可扩展的生物信息学工作流引擎。《Bioinformatics》，28(19):2520–2522，2012年。doi: 10.1093/bioinformatics/bts480。[54] P. Di Tommaso、M. Chatzou、E. W. Floden、P. P. Barja、E. Palumbo 和 C. Notredame。Nextflow实现可复现的计算工作流。《Nature Biotechnology》，35(4):316–319，2017年。doi: 10.1038/nbt.3820。[55] F. Mölder、K. P. Jablonski、B. Letcher、M. B. Hall、P. C. van Dyken、C. H. Tomkins-Tinch、V. Sochat、J. Forster、F. G. Vieira、C. Meesters 和 S. Lee。使用Snakemake进行可持续数据分析。F1000Research，10:33，2021年。doi: 10.12688/f1000research.29032.3。[56] S. Kanwal、F. Z. Khan、A. Lonie 和 R. O. Sinnott。探究可复现性并追踪溯源——基因组工作流案例研究。《BMC Bioinformatics》，18(1):337，2017年。doi: 10.1186/s12859-017-1747-0。[57] Virginia Braun 和 Victoria Clarke。心理学中的主题分析方法。《Qualitative Research in Psychology》，3(2):77–101，2006年。doi: 10.1191/1478088706qp063oa。[58]Anthropic。模型上下文协议规范，2024年。URL https://modelcontextprotocol.io/specification。访问日期：2025-06-01。[59] Philip A. Ewels、Alexander Peltzer、Sven Fillinger、Harshil Patel、Johannes Alneberg、Andreas Wilm、Maxime Ulysse Garcia、Paolo Di Tommaso 和 Sven Nahnsen。面向社区协作维护的生物信息学流水线的 nf-core 框架。《Nature Biotechnology》，38(3):276–278，2020年。doi: 10.1038/s41587-020-0439-x。[60] B. D. Conduit、N. G. Jones、H. J. Stone 和 G. J. Conduit。利用神经网络设计镍基高温合金。《Materials & Design》，131:358–365，2017年。doi: 10.1016/j.matdes.2017.06.007。