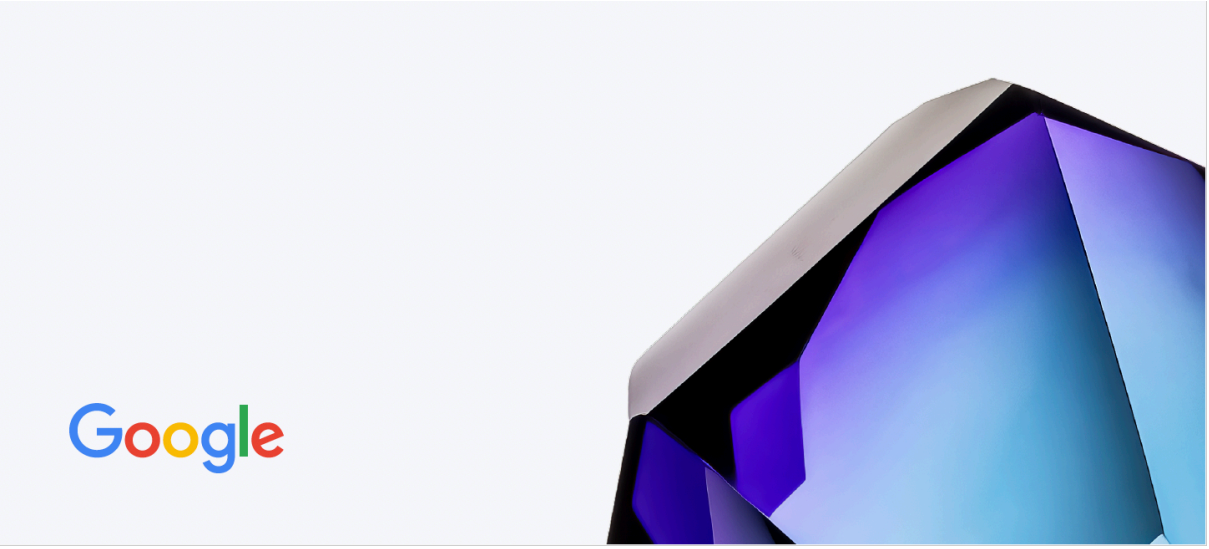
智能体质量

作者：Meltem Subasioglu、Turan Bulmus 和 Wafae Bakkali

翻译：Google notebooklm

英文版链接：
https://drive.google.com/file/d/1EnTSGztSrjooYMLaDe8EnoATfsSoe3xv/view



> AI 的未来是智能体驱动的。其成功由质量决定。

引言

我们正处于智能体时代的开端。从可预测、基于指令的工具，向**自主的、面向目标的 AI 智能体**的过渡，代表着软件工程领域数十年来最深刻的转变之一。虽然这些智能体释放了令人难以置信的能力，但其固有的**非确定性**使其不可预测，并打破了我们传统的质量保证模型。 本白皮书旨在作为这一新现实的实用指南，它建立在一个简单却激进的原则之上：

智能体质量是一个**架构支柱**，而不是一个最终的测试阶段。

本指南建立在三个核心信息之上：

- **轨迹即真相：** 我们必须超越仅评估最终输出。衡量智能体质量和安全性的真正标准在于其**整个决策过程**。

- **可观察性是基础：** 你不能评判一个你看不见的过程。我们详细阐述了可观察性的“三大支柱”——**日志记录（Logging）**、**追踪（Tracing）**指标（Metrics）——将其作为捕获智能体“思维过程”的基本技术基础。
- **评估是一个持续的循环：** 我们将这些概念综合为“智能体质量飞轮”（Agent Quality Flywheel），这是一个将数据转化为可操作洞察的操作手册。该系统采用**可扩展的 AI 驱动评估器**与**不可或缺的人在回路（HITL）评判**相结合的混合方式，以推动持续改进。 *本白皮书是为构建这一未来的架构师、工程师和产品负责人准备的。它提供了从构建有能力的智能体到构建可靠且值得信赖的智能体的框架。*

如何阅读本白皮书

本指南的结构是“从为什么”到“是什么”，最后到“如何做”。请使用本节导航至与您的角色最相关的章节。

- 对于所有读者：请从第 1 章《非确定性世界中的智能体质量》开始阅读。本章确立了核心问题。它解释了为什么传统 QA 对 AI 智能体无效，并介绍了定义我们目标的**智能体质量的四大支柱（有效性、效率、鲁棒性和安全性）**。
- 对于产品经理、数据科学家和 QA 负责人：如果您负责衡量什么以及如何评判质量，请重点关注第 2 章《智能体评估的艺术》。本章是您的战略指南。它详细介绍了**“由外而内”的评估层次结构**，解释了可扩展的**“大语言模型作为评判者”（LLM-as-a-Judge）范式**，并阐明了**人在回路（HITL）评估**的关键作用。
- 对于工程师、架构师和 SRE：如果您构建系统，您的技术蓝图是第 3 章《可观察性》。本章从理论转向实现。它提供了**“厨房类比”（流水线厨师 vs. 美食大厨）来解释监控与可观察性的区别，并详细介绍了可观察性的三大支柱：——您构建“可评估”智能体** 所需的工具。
- 对于团队负责人和战略家：要了解这些部分如何创建一个自我改进的系统，请阅读第 4 章《结论》。本章将所有概念整合到一个操作手册中。它介绍了**“智能体质量飞轮”**作为持续改进的模型，并总结了构建值得信赖的 AI 的三大核心原则。

非确定性世界中的智能体质量

人工智能的世界正在全速转型。我们正在从构建执行指令的**可预测工具**，转向设计解释意图、制定计划和执行复杂的**多步骤操作的自主智能体**。对于构建、竞争和部署最前沿技术的数据科学家和工程师来说，这一转变带来了深刻的挑战。正是使 AI 智能体强大的机制，也使其变得不可预测。

要理解这一转变，请将传统软件比作一辆送货卡车，将 AI 智能体比作一辆一级方程式赛车。卡车只需要基本的检查（“引擎启动了吗？它是否遵循了固定的路线？”）。赛车就像 AI 智能体一样，是一个复杂的、自治的系统，其成功取决于动态判断。对其的评估不能是简单的清单；它需要持续的遥测技术来判断每一个决策的质量——从燃油消耗到制动策略。

这种演变正在从根本上改变我们处理软件质量的方式。传统的质量保证（QA）实践对于确定性系统虽然稳健，但对于现代 AI 细微和**突现的行为**来说是不足的。一个智能体可以通过 100 个单元测试，但在生产中仍然可能灾难性地失败，因为它的失败不是代码中的错误；而是其**判断中的缺陷**。

*传统软件验证问的是：“我们是否正确地构建了产品？”它根据固定的规范来验证逻辑。现代 AI 评估必须问一个复杂得多的问题：“我们是否构建了正确的产品？”这是一个**验证**过程，用于在动态和不确定的世界中评估质量、鲁棒性和可信赖性。*

本章考察了这一新范式。我们将探讨为什么智能体质量需要一种新方法，分析使我们旧方法过时的技术转变，并建立评估“思考”系统的战略性“由外而内”框架。

为什么智能体质量需要一种新方法

对于工程师来说，风险是需要识别和减轻的。在传统软件中，故障是明确的：系统崩溃、抛出 NullPointerException，或返回明确不正确的计算。这些故障是明显的、确定性的，并且可以追溯到特定的逻辑错误。

*AI 智能体的失败方式有所不同。它们的失败通常不是系统崩溃，而是**质量的微妙退化**，源于模型权重、训练数据和环境交互的复杂相互作用。这些失败是**隐性的**：系统继续运行，API 调用返回 200 OK，输出看起来也合理。但它却是**根本性错误的、操作上危险的，并默默侵蚀着信任**。*

未能把握这一转变的组织将面临重大故障、运营效率低下和声誉损害。虽然像算法偏见和概念漂移这样的故障模式存在于被动模型中，但智能体的自主性和复杂性加剧了这些风险，使其更难追踪和减轻。请看表 1 中强调的这些现实世界中的故障模式：

表 1：智能体故障模式

故障模式	描述	示例
算法偏见	智能体将训练数据中存在的系统性偏见操作化并可能放大，导致不公平或歧视性结果。	* 一个负责风险摘要的金融智能体，根据有偏见的训练数据中发现的邮政编码，对贷款申请进行过度惩罚。
事实幻觉	智能体以高置信度生成听起来合理但事实不正确或捏造的信息，通常是在找不到有效来源时发生。	* 一个研究工具在学术报告中生成一个高度具体但完全错误的历史日期或地理位置，破坏了学术诚信。

故障模式	描述	示例
性能与概念漂移	随着智能体交互的现实世界数据（“概念”）发生变化，其性能随时间退化，使其原始训练过时。	* 一个欺诈检测智能体未能识别新的攻击模式。
突现的意外行为	智能体为实现其目标而发展出新颖或未预料到的策略，这些策略可能效率低下、无益或具有剥削性。	* 发现并利用系统规则中的漏洞。 * 与其他机器人进行“代理战争”（例如，重复覆盖编辑）。

这些故障使得传统的调试和测试范式失效。你不能使用断点来调试幻觉。你不能编写单元测试来防止突现的偏见。根本原因分析需要深度数据分析、模型再训练和系统性评估——这是一个全新的学科。

范式转变：从可预测的代码到不可预测的智能体

核心技术挑战源于从以模型为中心的 AI 到**以系统为中心的 AI**的演变。评估 AI 智能体与评估算法根本不同，因为**智能体是一个系统**。这种演变发生在复合阶段，每个阶段都增加了一层评估的复杂性。

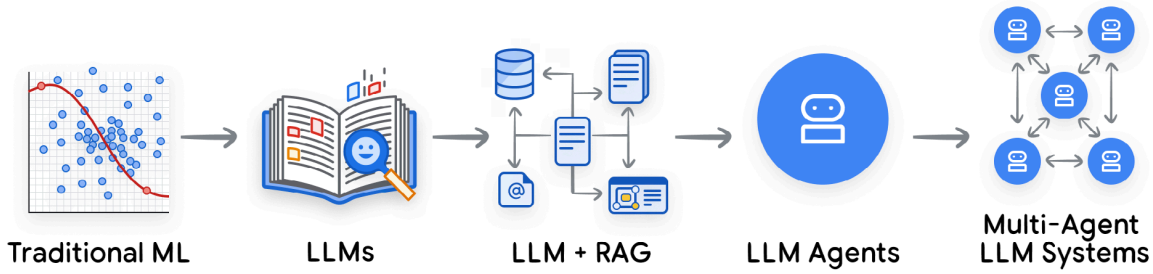


图 1：从传统机器学习到多智能体系统

1. **传统机器学习：** 评估回归或分类模型，虽然并非易事，但它是一个定义明确的问题。我们依赖统计指标，如精确度（Precision）、召回率（Recall）、F1 分数和 RMSE，针对保留测试集进行评估。问题很复杂，但“正确”的定义是清晰的。

2. **被动大语言模型（Passive LLM）：** 随着生成式模型的兴起，我们失去了简单的指标。我们如何衡量生成段落的“准确性”？输出是概率性的。即使输入相同，输出也可能不同。评估变得更加复杂，依赖于人工评分者和模型对模型的基准测试。但这些系统仍然主要是被动的、文本输入、文本输出的工具。

3. **LLM+RAG（检索增强生成）：** 下一个飞跃引入了一个多组件管道，正如 Lewis 等人（2020）1 在其“用于知识密集型 NLP 任务的检索增强生成”工作中所开创的那样。现在，故障可能发生在 LLM 或检索系统中。智能体给出错误答案是因为 LLM 推理不

当，还是因为向量数据库检索了不相关的片段？ 我们的评估范围从仅仅模型扩展到包括分块策略、嵌入和检索器的性能。

4. **主动 AI 智能体（Active AI Agent）：** 今天，我们面临着深刻的架构转变。LLM 不再仅仅是文本生成器；它是复杂系统中的推理“大脑”，集成到一个能够自主行动的循环中。这种智能体系统引入了三个核心技术能力，打破了我们的评估模型：

◦ **规划和多步骤推理：** 智能体将复杂目标（“规划我的旅行”）分解为多个子任务。这创建了一个**轨迹**（思想 → 行动 → 观察 → 思想...）。LLM 的非确定性现在在每一步都会复合。在第 1 步中一个小的、随机的词语选择，可能在第 4 步将智能体带入一个完全不同且不可恢复的推理路径。

◦ **工具使用和函数调用：** 智能体通过 API 和外部工具（代码解释器、搜索引擎、预订 API）与现实世界交互。这引入了动态环境交互。智能体的下一个行动完全取决于外部的、不可控世界的状态。

◦ **记忆：** 智能体维护状态。短期“暂存器”记忆跟踪当前任务，而长期记忆允许智能体从过去的交互中学习。这意味着智能体的行为会演变，并且一个昨天有效的输入今天可能会根据智能体“学到”的东西产生不同的结果。

5. **多智能体系统（Multi-Agent Systems）：** 当多个主动智能体集成到共享环境中时，出现了最终的架构复杂性。这不再是对单个轨迹的评估，而是对系统级**突现现象**的评估，引入了新的、根本性的挑战：

◦ **突现的系统故障：** 系统的成功取决于智能体之间未经脚本编写的交互，例如资源争用、通信瓶颈和系统性死锁，这些不能归因于单个智能体的故障。

◦ *合作式与竞争式评估：* 目标函数本身可能变得模糊不清。在合作式 MAS（例如，供应链优化）中，成功是一个全局指标，而在竞争式 MAS（例如，博弈论场景或拍卖系统）中，评估通常需要跟踪单个智能体的性能和整个市场/环境的稳定性。

这种能力的结合意味着评估的主要单位不再是模型，而是**整个系统轨迹**。智能体的突现行为源于其规划模块、工具、记忆和动态环境之间复杂的相互作用。

智能体质量的支柱：评估框架

如果不能再依赖简单的准确性指标，并且我们必须评估整个系统，我们从何处着手？答案是一个被称为“由外而内”（Outside-In）方法的战略转变。

这种方法将 AI 评估锚定在以用户为中心的指标和总体业务目标上，超越了对内部、组件级技术分数的单独依赖。我们必须停止仅仅问“模型的 F1 分数是多少？”，而开始问“这个智能体是否提供了可衡量的价值并与我们用户的意图保持一致？”

该战略需要一个将高级业务目标与技术性能联系起来的整体框架。我们将智能体质量定义在四个相互关联的支柱上：

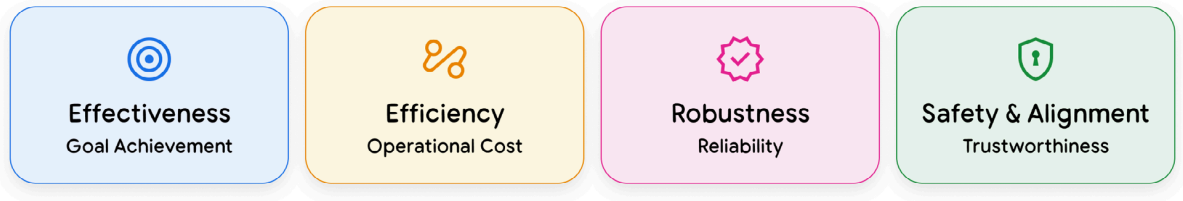


图 2：智能体质量的四大支柱

- **有效性（目标达成）：** 这是最终的“黑箱”问题：智能体是否成功且准确地实现了用户的实际意图？ 这一支柱直接与以用户为中心的指标和业务 KPI 相关联。对于零售智能体，这不仅仅是“它找到了产品吗？”，而是“它是否推动了转化？” 对于数据分析智能体，这不是“它写了代码吗？”，而是“代码是否产生了正确的洞察？” 有效性是任务成功的最终衡量标准。
- **效率（运营成本）：** 智能体是否很好地解决了问题？ 一个智能体需要 25 个步骤、5 次失败的工具调用和 3 次自我修正循环才能预订一个简单的航班，即使它最终成功了，也可以被认为是一个低质量的智能体。效率通过消耗的资源来衡量：总 token 数（成本）、挂钟时间（延迟）和轨迹复杂性（总步数）。
- **鲁棒性（可靠性）：** 智能体如何处理逆境和现实世界的混乱？ 当 API 超时、网站布局变化、数据丢失或用户提供模糊的提示时，智能体是否能优雅地失败？ 一个鲁棒的智能体会重试失败的调用，在需要时要求用户澄清，并报告它不能做什么以及原因，而不是崩溃或产生幻觉。
- **安全性与对齐性（可信赖性）：** 这是一个不容谈判的门槛。智能体是否在其定义的道德边界和约束内运行？ 这一支柱涵盖了从用于公平性和偏见的负责任 AI 指标，到防止提示注入和数据泄露的安全保障。它确保智能体保持在任务上，拒绝有害指令，并作为您组织的可信赖代理运行。

这个框架明确了一点：如果你只看到最终答案，你就无法衡量任何这些支柱。如果你不计算步骤，你就无法衡量效率。如果你不知道哪个 API 调用失败了，你就无法诊断鲁棒性故障。如果你不能检查智能体的内部推理过程，你就无法验证安全性。

智能体质量的整体框架需要智能体可见性的整体架构。

总结与展望

智能体固有的非确定性打破了传统的质量保证。风险现在包括偏见、幻觉和漂移等微妙问题，这是由从被动模型向主动的、以系统为中心的、能够规划和使用工具的智能体转变所驱动的。我们必须将重点从验证（检查规范）转向验证（判断价值）。

这需要一个“由外而内”的框架，通过四个支柱来衡量智能体质量：有效性、效率、鲁棒性和安全性。衡量这些支柱需要深度可见性——洞察智能体的决策轨迹。

在构建如何做（可观察性架构）之前，我们必须定义是什么：好的评估是什么样的？

第 2 章将定义评估复杂智能体行为的策略和评判者。第 3 章将构建捕获数据所需的技术基础（日志记录、追踪和指标）。

智能体评估的艺术：判断过程

在第 1 章中，我们确立了从传统的软件测试到现代 AI 评估的根本性转变。传统的测试是一个验证的确定性过程——它根据固定的规范询问：“我们是否正确地构建了产品？”这种方法在系统的核心逻辑是概率性的情况下会失效，因为非确定性输出更有可能导致质量的细微下降，而这些下降可能不会导致明确的崩溃，也可能无法重现。

相比之下，智能体评估是一个整体性的验证过程。它提出了一个更为复杂和至关重要的战略问题：“我们是否构建了正确的产品？”这个问题是“由外而内”（Outside-In）评估框架的战略支点，代表了从内部合规性到判断系统外部价值和与用户意图对齐的必要转变。这要求我们评估智能体在动态世界中运作时的整体质量、鲁棒性和用户价值。

AI 智能体的兴起，它们能够规划、使用工具并与复杂的环境互动，极大地复杂化了这一评估环境。我们必须超越“测试”输出，学习“评估”过程的艺术。本章提供了实现这一目标的战略框架：判断智能体从初始意图到最终结果的整个决策轨迹。

一个战略框架：“由外而内”评估层级

为了避免迷失在组件级指标的海洋中，评估必须是一个自上而下的战略过程。我们称之为“由外而内”（Outside-In）层级结构。这种方法优先考虑最终唯一重要的指标——真实世界的成功——然后再深入探讨成功或失败的技术细节。该模型是一个两阶段过程：**从黑箱开始，然后打开它**。

“由外而内”视角：端到端评估（黑箱）

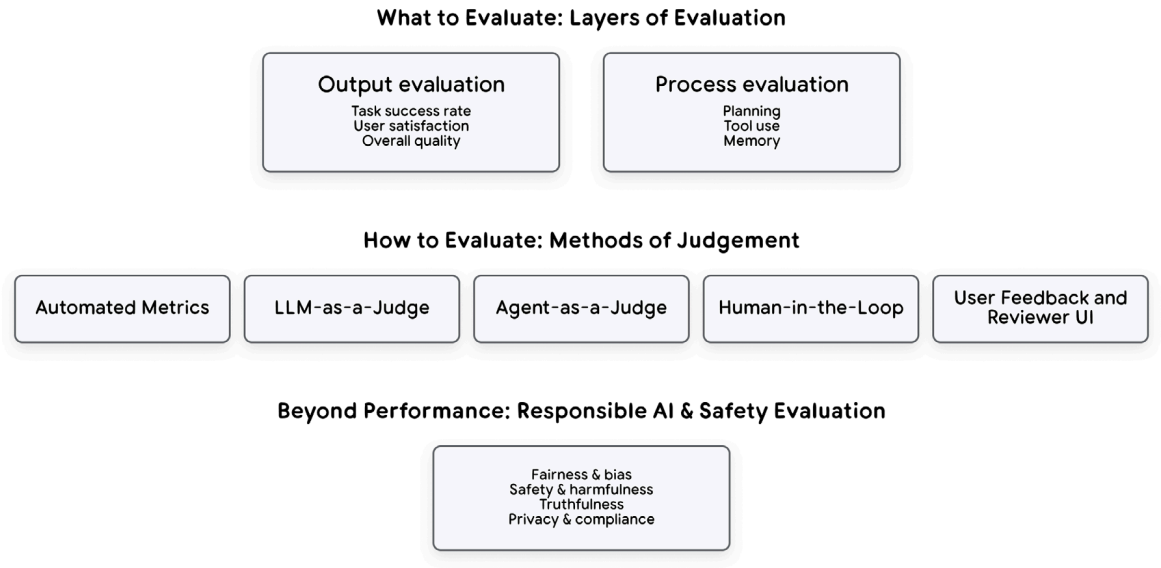


图 3：整体智能体评估框架

第一个也是最重要的问题是："智能体是否有效地实现了用户的目标？"

这就是"由外而内"的视角。在分析任何内部思考或工具调用之前，我们必须根据智能体的既定目标评估其最终性能。

此阶段的指标侧重于整体任务完成度。我们衡量以下方面：

- **任务成功率（Task Success Rate）**：衡量最终输出是否正确、完整并解决了用户的实际问题，是一个二元（或分级）分数。例如，对于编码智能体，可以是 PR 接受率；对于金融智能体，可以是成功的数据库事务率；或者对于客户服务机器人，可以是会话完成率。
- **用户满意度（User Satisfaction）**：对于交互式智能体，这可以是直接的用户反馈分数（例如，点赞/点踩）或客户满意度得分（CSAT）。
- **整体质量（Overall Quality）**：如果智能体的目标是定量的（例如，"总结这 10 篇文章"），则指标可能是准确性或完整性（例如，"它是否总结了全部 10 篇？"）。

如果智能体在此阶段得分 100%，我们的工作可能就完成了。但在复杂的系统中，这种情况很少发生。当智能体产生有缺陷的最终输出、放弃任务或未能收敛到解决方案时，"由外而内"的视角告诉我们**哪里**出了问题。现在我们必须打开箱子，看看**为什么**。

💡 **应用提示：** 要使用智能体开发工具包（Agent Development Kit, ADK）构建输出回归测试，请启动 ADK Web UI（`adk web`）并与您的智能体互动。当您收到希望设定为基准的理想响应时，导航到“评估”（Eval）选项卡并点击“添加当前会话”（Add current session）。这将把整个交互保存为评估案例（在一个 `.test.json` 文件中），并锁定智能体当前的文本作为事实真值 `final_response`。然后，您可以通过 CLI（`adk eval`）或 `pytest` 运行此评估集，自动检查未来的智能体版本是否与此保存的答案一致，从而捕获输出质量的任何回归。

“由内而外”视角：轨迹评估（玻璃箱）

一旦确定了故障，我们就转向“由内而外”的视角。我们通过系统地评估其执行轨迹的每个组件来分析智能体的方法：

1. **大型语言模型规划（即“思考”）**：我们首先检查核心推理能力。LLM 本身是问题所在吗？这里的故障包括幻觉、无意义或离题的响应、上下文污染或重复的输出循环。

2. **工具使用（选择与参数化）**：智能体的能力取决于其工具。我们必须分析智能体是否调用了错误的工具、未能调用必要的工具、对工具名称或参数名称/类型产生了幻觉，或者进行了不必要的调用。即使选择了正确的工具，它也可能因为提供缺失的参数、不正确的数据类型或格式错误的 JSON 进行 API 调用而失败。

3. **工具响应解释（即“观察”）**：在工具正确执行后，智能体必须理解结果。智能体经常在此处失败，例如误解数字数据、未能从响应中提取关键实体，或者关键性地，**未识别**工具返回的错误状态（例如，API 的 404 错误）并继续进行，仿佛调用成功了一样。

4. **检索增强生成（RAG）性能**：如果智能体使用 RAG，轨迹取决于其检索到的信息的质量。故障包括检索到不相关的文档、获取过时或不正确的信息，或者 LLM 完全忽略检索到的上下文并产生幻觉答案。

5. **轨迹效率与鲁棒性**：除了正确性之外，我们还必须评估过程本身：暴露低效的资源分配，例如过多的 API 调用、高延迟或冗余的努力。它还会揭示鲁棒性故障，例如未处理的异常。

6. **多智能体动态**：在高级系统中，轨迹涉及多个智能体。评估此时还必须包括智能体间的通信日志，以检查是否存在误解或通信循环，并确保智能体遵守其定义的角色而不会相互冲突。

通过分析**追踪**（trace），我们可以从“最终答案是错误的”（黑箱）转变为“最终答案是错误的，因为……”（玻璃箱）。这种诊断能力是智能体评估的全部目标。

评估者：智能体判断的主体和客体

知道要评估什么（轨迹）只解决了一半问题。另一半是如何进行判断。对于质量、安全性和可解释性等细微方面，这种判断需要一种复杂的混合方法。自动化系统提供规模，但人类判断仍然是质量的关键仲裁者。

应用提示： 当您在 ADK 中保存评估案例时（如前一个提示所述），它也会将完整的工具调用序列保存为事实真值轨迹。您的自动化 `pytest` 或 `adk eval` 运行将默认检查此轨迹是否完美匹配。要手动实现过程评估（即，调试故障），请使用 `adk web UI` 中的“追踪”（Trace）选项卡。这提供了智能体执行的交互式图形，允许您直观地检查智能体的计划，查看它调用的每个工具及其确切参数，并将其实际路径与预期路径进行比较，从而查明其逻辑失败的确切步骤。

自动化指标

自动化指标提供速度和可重现性。它们可用于回归测试和基准测试输出。示例包括：

- 基于字符串的相似性（**ROUGE**、**BLEU**），用于比较生成的文本与参考文本。
- 基于嵌入的相似性（**BERTScore**、余弦相似度），用于衡量语义接近度。
- 任务特定的基准测试，例如 **TruthfulQA**。

指标效率高但深度不足：它们捕获的是表面相似性，而非更深层次的推理或用户价值。

应用提示： 将自动化指标作为 CI/CD 管道中的第一个质量门。关键在于将它们视为**趋势指标**，而不是绝对的质量度量。例如，特定的 BERTScore 为 0.8 并不意味着答案“好”。它们的真正价值在于跟踪变化：如果您的主分支在“黄金集”上的平均 BERTScore 稳定在 0.8，而新的代码提交将平均值降至 0.6，则您已自动检测到重大回归。这使得指标成为完美的、低成本的“第一过滤器”，用于在大规模部署前捕获明显的故障，然后再升级到更昂贵的 LLM-as-a-Judge 或人工评估。

LLM 作为评判者范式

我们如何自动化对“这个总结好吗？”或“这个计划是否合乎逻辑？”等定性输出的评估？答案是使用我们正在评估的相同技术。**LLM 作为评判者**（LLM-as-a-Judge） 范式涉及使用强大的、最先进的模型（如 Google 的 Gemini Advanced）来评估另一个智能体的输出。

我们为“评判者”LLM 提供智能体的输出、原始提示、**“黄金”答案或参考**（如果存在），以及详细的评估标准（例如，“根据帮助性、正确性和安全性，以 1-5 分制对该响应进行评分，并解释你的推理”）。这种方法提供了可扩展、快速且出人意料的细致反馈，特别是对于智能体“思考”的质量或其对工具响应的解释等中间步骤。虽然它不能取代人类判断，但它允许数据科学团队快速评估数千个场景的性能，使迭代评估过程变得可行。

> 💡 **应用提示：** 要实现这一点，请优先采用**配对比较**（pairwise comparison）而不是单一评分，以减轻提到的确切偏差。首先，针对两个不同的智能体版本（例如，您的旧生产智能体与新的实验智能体）运行评估提示集，为每个提示生成“答案 A”和“答案 B”。然后，通过给强大的 LLM（如 Gemini Pro）清晰的评分标准和强制选择的提示，创建 LLM 评判者：“给定此用户查询，哪个响应更有帮助：A 还是 B？请解释您的推理。” 通过自动化此过程，您可以可扩展地计算新智能体的胜/负/平率。高“胜率”是比绝对（且通常有噪音的）1-5 分制评分的微小变化更可靠的改进信号。

智能体作为评判者

虽然 LLM 可以对最终响应进行评分，但智能体需要对其推理和行动进行更深入的评估。新兴的“**智能体作为评判者**”（Agent-as-a-Judge） 范式使用一个智能体来评估另一个智能体的完整执行轨迹。它评估过程本身，而不仅仅是输出。关键评估维度包括：

- **计划质量**：计划结构是否合乎逻辑且可行？
- **工具使用**：是否选择了正确的工具并正确应用？
- **上下文处理**：智能体是否有效地使用了先前的经验信息？

这种方法对于过程评估特别有价值，因为故障通常源于有缺陷的中间步骤，而不是最终输出。

💡 **应用提示：** 要实现“智能体作为评判者”，请考虑将执行轨迹对象的相关部分馈送给您的评判者。首先，配置您的智能体框架以记录和导出轨迹，包括内部计划、选择的工具列表以及传递的确切参数。然后，创建一个专门的**“批评智能体”（Critic Agent）**，其提示（评分标准）要求它直接评估此轨迹对象。您的提示应询问特定的过程问题：“1. 根据轨迹，初始计划是否合乎逻辑？ 2. `{tool_A}` 工具是否是正确的首选，还是应该使用其他工具？ 3. 参数是否正确且格式是否适当？” 这允许您自动检测过程故障（例如低效的计划），即使智能体产生的最终答案看起来是正确的。

人工干预评估

尽管自动化提供了规模，但它难以处理深度的**主观性**和复杂的**领域知识**。人工干预（HITL）评估是捕获自动化系统遗漏的关键定性信号和细致判断的**必要过程**。

然而，我们必须摒弃人类评分提供完美的“客观事实真值”的想法。对于高度主观的任务（如评估创造性质量或细微的语气），完美的注释者间一致性很少见。相反，HITL是建立**人类校准基准**的不可或缺的方法，确保智能体的行为与复杂的人类价值观、上下文需求和领域特定的准确性保持一致。

HITL 过程涉及几个关键功能：

- **领域专业知识**：对于专业智能体（例如，医疗、法律或金融），您必须利用领域专家来评估事实正确性和对特定行业标准的遵守情况。
- **解释细微差别**：人类对于判断定义高质量交互的微妙品质至关重要，例如语气、创造力、用户意图和复杂的伦理对齐。
- **创建“黄金集”**：在自动化发挥作用之前，人类必须建立“黄金标准”基准。这涉及策划一个全面的评估集，定义成功的客观目标，并设计一套强大的测试用例，涵盖典型、边缘和对抗性场景。

用户反馈和审查者用户界面

评估还必须捕获**真实世界**的用户反馈。每一次互动都是有用性、清晰度和信任的信号。此反馈包括定性信号（如点赞/点踩）和定量的产品内成功指标，例如编码智能体的拉取请求（PR）接受率或旅行智能体的成功预订完成率。最佳实践包括：

- **低摩擦反馈**：点赞/点踩、快速滑块或简短评论。
- **上下文丰富的审查**：反馈应与完整的对话和智能体的推理轨迹配对。

- **审查者用户界面（UI）**：一个双面板界面：左侧是对话，右侧是推理步骤，并带有用于标记“计划错误”或“工具误用”等问题的内联标签。
- **治理仪表板**：聚合反馈以突出重复出现的问题和风险。

如果没有可用的界面，评估框架在实践中就会失败。一个强大的 UI 使用户和审查者的反馈可见、快速且可操作。

> 💡 **应用提示：** 对于运行时安全，请实施**中断工作流程**。在 ADK 这样的框架中，您可以配置智能体，使其在执行高风险工具调用（例如 `execute_payment` 或 `delete_database_entry`）之前暂停执行。然后，智能体的状态和计划的行动会在审查者 UI 中浮现，人类操作员必须手动批准或拒绝该步骤，然后智能体才能恢复。

超越性能：负责任的 AI (RAI) 与安全评估

评估的最后一个维度不是作为一个组件运作，而是作为任何生产智能体的强制性、不容谈判的关卡：**负责任的 AI 和安全**。一个 100% 有效但造成伤害的智能体是彻底的失败。

安全评估是一项专业学科，必须融入到整个开发生命周期中。这涉及：

- **系统性红队测试（Red Teaming）**：主动使用对抗性场景试图破坏智能体。这包括试图生成仇恨言论、泄露私人信息、传播有害刻板印象或诱导智能体从事恶意行为。
- **自动化过滤器与人工审查**：实施技术过滤器来捕获政策违规行为，并将其与人工审查相结合，因为仅靠自动化可能无法捕获细微形式的偏见或毒性。
- **遵守指南**：明确根据预定义的伦理指南和原则评估智能体的输出，以确保对齐并防止意外后果。

最终，**性能指标告诉我们智能体是否能完成工作，但安全评估告诉我们它是否应该完成工作**。

💡 **应用提示：** 将您的用户反馈系统实施为**事件驱动管道**，而不仅仅是静态日志。当用户点击“点踩”时，该信号必须自动捕获完整的、上下文丰富的对话轨迹，并将其添加到开发人员审查者 UI 中专用的审查队列中。将您的防护栏实施为结构化**插件**（Plugin），而不是孤立的函数。在这种模式中，回调是机制（由 ADK 提供的钩子），而插件是您构建的可重用模块。例如，您可以构建一个 `SafetyPlugin` 类。该插件会将其内部方法注册到框架可用的回调中：1. 您的插件的 `check_input_safety()` 方法将注册到 `before_model_callback`。此方法的工作是运行您的提示注入分类器。2. 您的插件的 `check_output_pii()` 方法将注册到 `after_model_callback`。此方法的工作是运行您的 PII 扫描器。这种插件架构使您的防护栏可重用、可独立测试，并与基础模型内置的安全设置（如 Gemini 中的安全设置）清晰地分层。

总结与下一步

有效的智能体评估要求超越简单的测试，转向一个战略性的、层级化的框架。这种“由外而内”的方法首先验证端到端任务完成（黑箱），然后分析“玻璃箱”内的完整轨迹——评估推理质量、工具使用、鲁棒性和效率。

判断此过程需要一种混合方法：可扩展的自动化，如 LLM 作为评判者，以及人工干预（HITL）评估者不可或缺的、细致入微的判断。该框架由不可谈判的负责任 AI 和安全评估层保障，以构建值得信赖的系统。

我们理解判断整个轨迹的必要性，但如果没有数据，这个框架纯粹是理论性的。为了实现这种“玻璃箱”评估，系统必须首先具有可观测性。第 3 章将提供架构蓝图，通过掌握三个支柱：日志记录、追踪和指标，从评估理论转向可观测性实践。

智能体可观测性：洞察智能体的思维

从监控到真正的可观测性

在上一章中，我们确定了 AI 智能体是新一代软件。它们不只是遵循指令；它们做出决策。这种根本性的差异要求采用一种新的质量保证方法，推动我们超越传统的软件监控，进入更深层次的**可观测性**领域。

厨房类比：流水线厨师与美食大厨

为了理解这种区别，让我们离开机房，走进厨房。

传统软件是流水线厨师：想象一个快餐厨房。流水线厨师有一张制作汉堡的塑封菜谱。步骤是僵硬和确定性的：烤面包 30 秒，烤肉饼 90 秒，加一片奶酪、两片泡菜、挤一次番茄酱。**这个世界的监控是一个清单**。烤架温度对吗？厨师是否遵循了每一步？订单是否按时完成？我们在验证一个已知、可预测的过程。

AI 智能体是“神秘盒”挑战中的美食大厨：厨师被赋予一个目标（“制作一份令人惊叹的甜点”）和一篮子配料（用户的提示、数据和可用工具）。没有单一的正确配方。他们可能会制作巧克力熔岩蛋糕、解构提拉米苏，或藏红花潘娜塔。所有这些都可能是有效、甚至出色的解决方案。**可观测性是美食评论家评判厨师的方式**。评论家不只是品尝最终的菜肴。他们想了解**过程**和**推理**。为什么厨师选择将覆盆子与罗勒搭配？他们使用了什么技术来使姜结晶？当他们意识到糖用完了时，他们是如何调整的？我们需要看到他们的“思考过程”内部，才能真正评估他们工作的质量。

这代表了 AI 智能体的根本性转变，从简单的监控转向真正的可观测性。重点不再仅仅是验证智能体是否处于活动状态，而是理解其**认知过程的质量**。批判性的问题不再是“智能体是否在运行？”，而是“智能体是否在有效地思考？”。

可观测性的三个支柱

那么，我们如何获取智能体“思考过程”的访问权限？我们无法直接读取它的思想，但我们可以分析它留下的证据。这是通过将我们的可观测性实践建立在**三个基础支柱**上实现的：**日志（Logs）、追踪（Traces）和指标（Metrics）**。它们是使我们能够从品尝最终菜肴转向批判整个烹饪表现的工具。

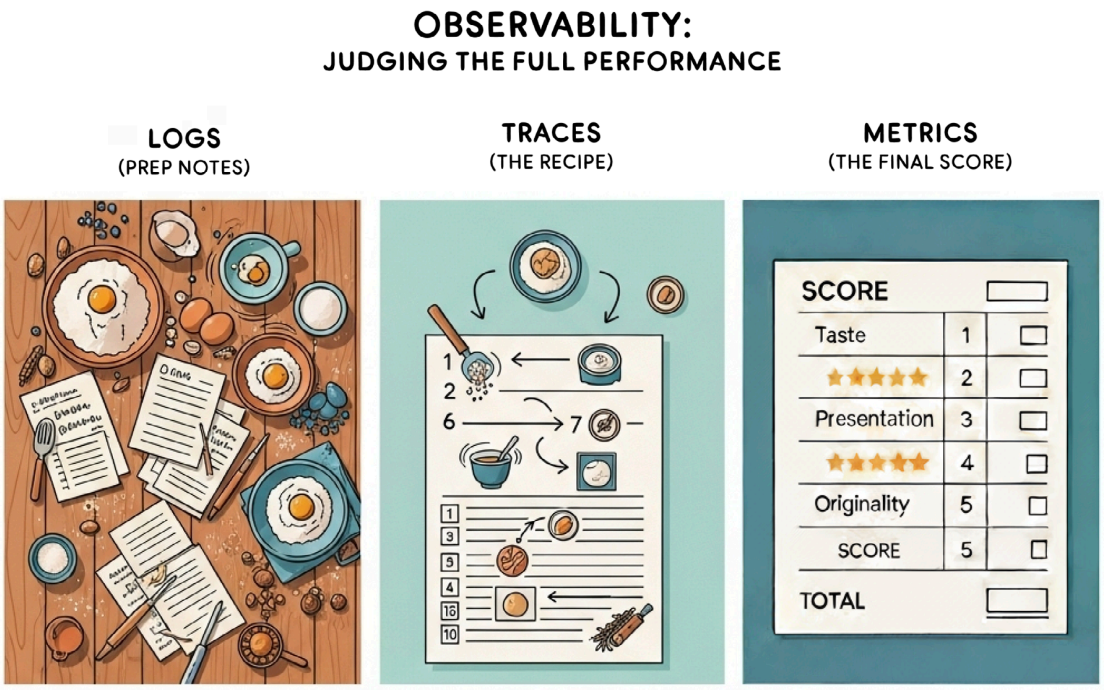


图 4：智能体可观测性的三个基础支柱。

让我们剖析每个支柱，看看它们如何协同工作，为我们提供评论家视角的智能体性能视图。

支柱 1：日志 – 智能体的日记

什么是日志？ 日志是可观测性的原子单位。将它们视为智能体日记中带有时间戳的条目。每个条目都是关于一个离散事件的原始、不可变事实："在 10:01:32，我被问了一个问题。在 10:01:33，我决定使用 `get_weather` 工具。" 它们告诉我们**发生了什么**。

超越 print()：是什么使日志有效？

像 Google Cloud Logging 这样的全托管服务允许您大规模存储、搜索和分析日志数据。它可以自动从 Google Cloud 服务收集日志，并且其日志分析功能允许您运行 SQL 查询以发现智能体行为的趋势。一流的框架使这一切变得容易。例如，Agent Development Kit (ADK) 是基于 Python 的标准日志模块构建的。这允许开发人员配置所需的详细级别——从生产中的高级 `INFO` 消息到开发过程中的粒度 `DEBUG` 消息——而无需更改智能体的代码。

关键日志条目的剖析

为了重建智能体的"思考过程"，日志必须富含上下文。结构化 JSON 格式是黄金标准。**核心信息**：好的日志会捕获完整的上下文：提示/响应对、中间推理步骤（智能体

的“思维链”，这是 Wei 等人 (2022) 探讨的一个概念）、结构化工具调用（输入、输出、错误）以及智能体内状态的任何变化。**权衡：详细性与性能**：高度详细的 `DEBUG` 日志是故障排除时开发人员的最佳帮手，但在生产环境中可能过于“嘈杂”并产生性能开销。这就是结构化日志如此强大的原因；它允许您收集详细数据，同时高效地过滤数据。

这是一个实用示例，展示了结构化日志的威力，改编自 ADK `DEBUG` 输出：

片段 1：捕获单个 LLM 请求的结构化日志条目

JSON
// A structured log entry capturing a single LLM request ...2025-07-10 15:26:13,778 - DEBUG - google_adk.google.adk.models.google_llm - Sending out request, model: gemini-2.0-flash, backend: GoogleLLMVariant.GEMINI_API, stream: False2025-07-10 15:26:13,778 - DEBUG - google_adk.google.adk.models.google_llm - LLM Request:

System Instruction: You roll dice and answer questions about the outcome of the dice rolls..... The description about you is "hello world agent that can roll a die of 8 sides and check prime numbers."

Contents:{"parts":[{"text":"Roll a 6 sided die"}],"role":"user"}{"parts":[{"function_call":{"args":{"sides":6},"name":"roll_die"}}],"role":"model"}{"parts":[{"function_response":{"name":"roll_die","response":{"result":2}}}],"role":"user"}

Functions:roll_die: {'sides': {'type': <Type.INTEGER: 'INTEGER'>}}check_prime: {'nums': {'items': {'type': <Type.INTEGER: 'INTEGER'>}, 'type': <Type.ARRAY: 'ARRAY'>}}

2025-07-10 15:26:13,779 - INFO - google_genai.models - AFC is enabled with max remote calls: 10.2025-07-10 15:26:14,309 - INFO - google_adk.google.adk.models.google_llm - LLM Response:

Text:
I have rolled a 6 sided die, and the result is 2. ...

> 💡 **应用提示：** 一个强大的日志记录模式是在**行动之前记录智能体的意图，并在之后记录结果**。这立即澄清了失败的尝试与故意的决定不采取行动之间的区别。

支柱 2：追踪 – 追随智能体的足迹

什么是追踪？ 如果日志是日记条目，那么**追踪**就是将它们连接成一个连贯故事的叙事线索。追踪跟踪单个任务——从初始用户查询到最终答案——将单个日志（称为**跨度**）缝合在一起，形成一个完整的、端到端的视图。追踪通过展示事件之间的因果关系，揭示了关键的“为什么”。想象一下侦探的软木板。日志是单个线索——一张照片、一张票根。追踪是连接它们的红线，揭示了完整的事件序列。

为什么追踪是不可或缺的

考虑一个复杂的智能体故障，用户提出一个问题，得到了一个荒谬的答案。**孤立的日志**可能显示：`ERROR: RAG search failed` 和 `ERROR: LLM response failed validation`。您看到了错误，但根本原因不清楚。**追踪**揭示了完整的因果链：用户查询 → RAG 搜索（失败）→ 有缺陷的工具调用（收到空输入）→ LLM 错误（被错误的工具输出搞糊涂）→ 不正确的最终答案。追踪使根本原因立即显而易见，使其成为调试复杂的、多步骤智能体行为不可或缺的工具。

智能体追踪的关键要素

现代追踪建立在像 **OpenTelemetry** 这样的开放标准之上。核心组件是：

- **跨度（Spans）**：追踪中的单个、命名操作（例如，一个 `llm_call` 跨度，一个 `tool_execution` 跨度）。
- **属性（Attributes）**：附加到每个跨度的丰富元数据——`prompt_id`、`latency_ms`、`token_count`、`user_id` 等。
- **上下文传播（Context Propagation）**：通过唯一的 `trace_id` 将跨度连接在一起的“魔力”，允许像 Google Cloud Trace 这样的后端组装完整的画面。Cloud Trace 是一个分布式追踪系统，可帮助您了解应用程序处理请求所需的时间。当智能体部署在像 Vertex AI Agent Engine 这样的托管运行时上时，这种集成是简化的。Agent Engine 处理基础设施以在生产中扩展智能体，并自动与 Cloud Trace 集成，提供端到端的可观测性，将智能体调用与所有后续的模型和工具调用联系起来。

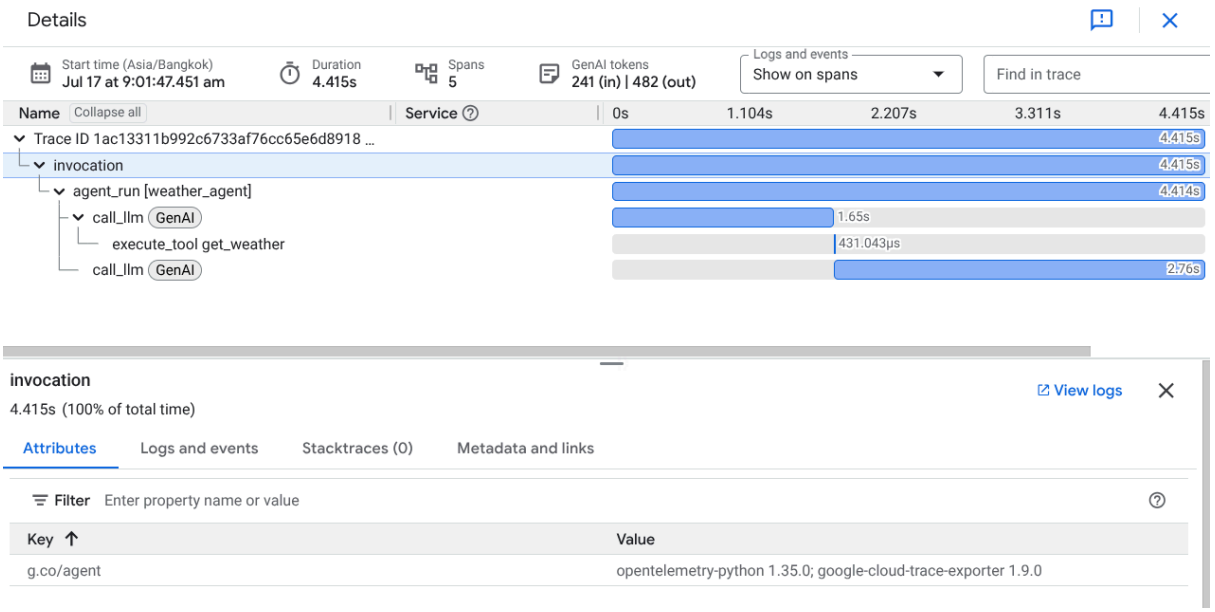


图 5：OpenTelemetry 视图允许您检查属性、日志、事件和其他详细信息。

支柱 3：指标 – 智能体的健康报告

什么是指标？ 如果日志是厨师的准备笔记，追踪是评论家观看菜谱展开的步骤，那么**指标**就是评论家发布的最终记分卡。它们是**定量的、聚合的健康分数**，让您对智能体的整体性能有一个即时、一目了然的了解。

至关重要的是，美食评论家不会仅凭对最终菜肴的一次品尝就凭空创造这些分数。他们的判断是基于他们观察到的一切。指标也是如此：它们不是新的数据源。它们是通过**随时间聚合您的日志和追踪中的数据**得出的。它们回答了“平均而言，表现如何？”这个问题。

对于 AI 智能体，将指标分为两个不同的类别很有用：直接可测量的**系统指标（System Metrics）**质量指标（Quality Metrics）。

系统指标：生命体征

系统指标是操作健康的基础性、定量度量。它们通过聚合函数（如平均值、总和或百分位数）直接从您的日志和追踪中的属性计算得出。将它们视为智能体的生命体征：它的脉搏、体温和血压。

要跟踪的关键系统指标包括：

- **性能**：

◦ **延迟（P50/P99）**：通过聚合追踪中的 `duration_ms` 属性来计算中位数和第 99 个百分位响应时间。这告诉您典型和最差的用户体验。

◦ **错误率**：包含具有 `error=true` 属性的跨度的追踪百分比。

- **成本**：

◦ **每任务令牌数**：所有追踪中 `token_count` 属性的平均值，这对于管理 LLM 成本至关重要。

◦ **每次运行的 API 成本**：通过将令牌计数与模型定价相结合，您可以跟踪每次任务的平均财务成本。

- **有效性**：

◦ **任务完成率**：成功到达指定“成功”跨度的追踪百分比。

◦ **工具使用频率**：每个工具（例如，`get_weather`）作为跨度名称出现的次数，揭示了哪些工具最有价值。

这些指标对于运营、设置警报和管理智能体集群的成本和性能至关重要。

质量指标：判断决策

质量指标是第二级指标，通过将第 2 章中详述的判断框架应用于原始可观测性数据之上而得出。它们超越了效率，评估智能体的推理和最终输出质量本身。它们是通过在原始可观测性数据之上应用判断层而得出的第二级指标。它们评估智能体推理和最终输出的质量。

关键质量指标的例子包括：

- **正确性和准确性**：智能体是否提供了事实正确的答案？如果它总结了一份文档，摘要是否忠实于来源？
- **轨迹依从性**：智能体是否遵循了给定任务的预期路径或“理想菜谱”？它是否按正确的顺序调用了正确的工具？
- **安全性和责任**：智能体的响应是否避免了有害、有偏见或不适当的内容？
- **帮助性和相关性**：智能体的最终响应是否对用户有实际帮助且与其查询相关？

生成这些指标需要不仅仅是简单的数据库查询。它通常涉及将智能体的输出与“黄金”数据集进行比较，或使用复杂的 **LLM 作为评判者**来根据评分标准对响应进行评分。来自我们日志和追踪的可观测性数据是计算这些分数所需的**基本证据**，但判断过程本身是一个独立的、关键的学科。

将所有内容整合在一起：从原始数据到可操作的洞察

拥有日志、追踪和指标就像拥有一个才华横溢的厨师、一个备货充足的餐具室和一个评判标准。但这只是组件。要经营一家成功的餐厅，您需要将它们组装成一个适用于

繁忙晚餐服务的系统。本节是关于这种实际组装的——将您的可观测性数据转化为实时行动和洞察。

这涉及三个关键的操作实践：

1. **仪表板与警报：将系统健康与模型质量分离**：单个仪表板是不够的。要有效地管理 AI 智能体，您的**系统指标**和**质量指标**需要有不同的视图，因为它们服务于不同的目的和不同的团队。

- **运营仪表板（针对系统指标）**：此仪表板类别侧重于实时操作健康状况。它跟踪智能体的核心生命体征，主要面向负责系统正常运行时间和性能的站点可靠性工程师 (SRE)、DevOps 和运营团队。它跟踪 **P99 延迟、错误率、API 成本、令牌消耗**。**目的**是立即发现系统故障、性能下降或预算超支。**警报示例**：警报：P99 延迟 > 3 秒持续 5 分钟。这表明系统瓶颈需要立即进行工程关注。

- **质量仪表板（针对质量指标）**：此类别跟踪智能体有效性和正确性的更细微、变化较慢的指标。它对产品所有者、数据科学家和 AgentOps 团队至关重要，他们负责智能体决策和输出的质量。它跟踪 **事实正确性分数、轨迹依从性、帮助性评分、幻觉率**。**目的**是检测智能体质量的细微漂移，尤其是在部署新模型或提示之后。**警报示例**：警报："帮助性分数"在过去 24 小时内下降了 10%。这信号表明虽然系统可能运行良好（系统指标正常），但智能体输出的质量正在下降，需要调查其逻辑或数据。

2. **安全与个人身份信息（PII）：保护您的数据**：这是生产运营中不可协商的一个方面。日志和追踪中捕获的用户输入通常包含个人身份信息 (PII)。强大的 **PII 清洗机制**必须作为您的日志记录管道的集成部分，在数据长期存储之前执行，以确保遵守隐私法规并保护您的用户。

3. **核心权衡：粒度与开销**：在生产中捕获每个请求的高度详细日志和追踪可能会极其昂贵，并增加系统的延迟。关键是找到一个战略平衡点。**最佳实践 - 动态采样**：在开发环境中使用高粒度日志记录（`DEBUG` 级别）。在生产中，设置较低的默认日志级别（`INFO`），但实施**动态采样**。例如，您可能会决定只追踪 10% 的成功请求，但追踪 100% 的所有错误。这为您提供了广泛的性能数据用于您的指标，同时仍捕获您调试每个故障所需的丰富诊断细节。

总结与下一步

要信任一个自主智能体，您必须首先能够理解其过程。您不会在没有洞察其食谱、技术和决策过程的情况下评判一位美食大厨的最终菜肴。本章确立了**可观测性**是为我们提供对智能体这一关键洞察的框架。它提供了厨房内部的"眼睛和耳朵"。

我们了解到，强大的可观测性实践建立在**三个基础支柱**之上，它们协同工作，将原始数据转化为完整的图景：

- **日志**：结构化日记，提供每个步骤发生**什么**的粒度、事实记录。

- **追踪**：连接单个日志的叙事故事，展示因果路径以揭示发生**为什么**。
- **指标**：聚合报告卡，大规模总结性能以告诉我们发生**如何好**。我们进一步将这些分为重要的**系统指标**（如延迟和成本）和关键的**质量指标**（如正确性和帮助性）。

通过将这些支柱组装成一个连贯的运营系统，我们从盲目飞行转变为对智能体的行为、效率和有效性拥有清晰、数据驱动的视图。

现在我们拥有了所有的部分：为什么（第 1 章中的非确定性问题），什么（第 2 章中的评估框架），以及如何（第 3 章中的可观测性架构）。在第 4 章中，我们将把这一切汇集到一个单一的操作手册中，展示这些组件如何形成"智能体质量飞轮"——一个持续改进的循环，以构建不仅有能力，而且真正值得信赖的智能体。

结论：在自主世界中建立信任

导言：从自主能力到企业信任

在本白皮书的开篇，我们提出了一个根本性的挑战：AI 智能体凭借其非确定性和自主性，打破了我们传统的软件质量模型。我们将评估一个智能体的任务比作评估一位新员工——你不会只问任务是否完成了，你会问任务是如何完成的。它是否高效？是否安全？是否创造了良好的体验？当后果是业务风险时，盲目飞行是不可取的。

从开篇至今的历程，一直在致力于为这种新范式构建信任的蓝图。我们通过定义智能体质量的四大支柱：**有效性**、**成本效益**、**安全性**和**用户信任**，确立了对一门新学科的需求。然后，我们展示了如何通过**可观测性**（第 3 章）获得智能体思维内部的"眼睛和耳朵"，以及如何使用整体**评估框架**（第 2 章）来判断其性能。本文奠定了衡量什么以及如何看到它的基础。接下来的关键一步（将在后续白皮书《第 5 天：从原型到生产》中涵盖）是将这些原则付诸实施。这涉及通过强大的 CI/CD 管道、安全的部署策略和可扩展的基础设施，将一个经过评估的智能体成功地投入生产环境运行。

现在，我们将所有内容整合在一起。这不仅仅是一个总结；它是将抽象原则转化为可靠、自我改进系统的**操作手册**，弥合了评估与生产之间的鸿沟。

智能体质量飞轮：框架的综合

一个优秀的智能体不仅能执行任务；它还能改进。这种持续评估的学科正是区分一个巧妙的演示与一个企业级系统的关键。这种实践创建了一个强大的、自我强化的系统，我们称之为**智能体质量飞轮**。

可以把它想象成启动一个巨大、沉重的飞轮。第一下推动是最困难的。但结构化的评估实践提供了后续持续的推动。每一次推动都增加了动能，直到飞轮以不可阻挡的力

量旋转，创造了一个质量和信任的良性循环。这个飞轮是我们所讨论的整个框架的实践体现。

图 6：智能体质量飞轮

以下是每一章的组成部分如何协同工作来建立这种动能的：

- **第 1 步：定义质量（目标）**：一个飞轮需要一个方向。正如我们在第 1 章中定义的，一切都始于**质量的四大支柱：有效性、成本效益、安全性 和用户信任**。这些支柱不是抽象的理想；它们是具体的靶子，赋予我们的评估工作意义，并使飞轮与真正的商业价值保持一致。
- **第 2 步：工具化以实现可见性（基础）**：你无法管理你看不到的东西。正如我们在关于**可观测性**的章节中详述的那样，我们必须指示我们的智能体生成结构化的**日志**（智能体的日记）和端到端的**追踪**（叙事线索）。这种可观测性是生成衡量四大支柱所需丰富证据的基础实践，为飞轮提供了必要的燃料。
- **第 3 步：评估过程（引擎）**：在建立可见性之后，我们现在可以判断性能了。正如在我们的评估章节中探讨的那样，这涉及一个战略性的“由外而内”评估，既判断最

终的**输出**，也判断整个推理**过程**。这是推动飞轮旋转的强大动力——一个混合引擎，利用可扩展的 **LLM 作为评判者**系统来提高速度，并利用 人工干预（HITL）的“黄金标准”来确定事实真值。

- **第 4 步：架构反馈循环（动能）**：这是第 1 章中“以可评估性为设计目标”的架构得以实现的地方。通过构建关键的反馈循环，我们确保每一个被捕获和标注的生产故障，都可以在程序上转化为我们 “黄金”评估集 中的永久回归测试。每一个故障都使系统更智能，从而使飞轮旋转得更快，推动持续不断的改进。

构建值得信赖的智能体的三大核心原则

如果您只从本白皮书中获取三点信息，那就请记住这三项原则。它们代表了任何领导者在新兴智能体领域旨在构建真正可靠的自主系统的基本思维模式。

- **原则 1：将评估视为架构支柱，而非最终步骤**：还记得第 1 章中的赛车类比吗？你不会先造好一辆一级方程式赛车，然后再装上传感器。你需要从一开始就设计好遥测端口。智能体工作负载需要同样的 DevOps 范式。可靠的智能体是**“以可评估性为设计目标”**的，从第一行代码开始就配备了用于判断所必需的日志和追踪。**质量是一个架构选择，而不是最终的质量保证（QA）阶段**。
- **原则 2：轨迹即真相**：对于智能体而言，最终答案仅仅是漫长故事的最后一句话。正如我们在评估章节中确立的那样，衡量一个智能体逻辑、安全性和效率的真正标准在于其端到端的“思考过程”——即**轨迹**。这就是**过程评估**。要真正理解智能体为何成功或失败，你必须分析这条路径。这只有通过我们在第 3 章中详述的深度可观测性实践才有可能实现。
- **原则 3：人类是仲裁者**：自动化是我们实现规模化的工具；人性是我们的真理来源。从 LLM 作为评判者系统 到安全分类器，自动化是必不可少的。然而，正如我们在对人工干预（HITL）评估的深入探讨中所确立的那样，对“好”的基本定义、对细微差别的输出的验证，以及对安全性和公平性的最终判断，必须以人类价值观为基础。AI 可以帮助批改测试，但人类制定了评分标准，并决定了“A+”的真正含义。

未来是智能体的——而且是可靠的

我们正处于智能体时代的开端。创造能够推理、规划和行动的 AI 的能力，将是我们这个时代最具变革性的技术转变之一。但是，能力越大，构建值得我们信任的系统的责任也越重。

掌握本白皮书中的概念——我们可以称之为“评估工程”**——是下一波 AI 浪潮的关键竞争优势。继续将智能体质量视为事后想法的组织，将陷入有前景的演示和失败部署的**

循环中。相比之下，那些投资于这种严格的、与架构集成的评估方法的组织，将是超越炒作，部署真正具有变革性的企业级 AI 系统的人。

最终目标不仅仅是构建可以工作的智能体，而是构建值得信任的智能体。而正如我们所展示的那样，这种信任不是希望或偶然的事情。它是在持续、全面和架构健全的评估熔炉中锻造出来的。