

亚马逊云科技



中国峰会

2026年6月23日-24日 上海 · 世博中心

Agent 黑盒拆解术： 基于 Langfuse 的 Trace、Token、Tool Call 可观测

Agent 使用是大势所趋，管控迫在眉睫

Uber's AI bill got so high that it is now limiting Claude Code usage to \$1,500: Report

Uber has set a \$1,500 monthly spending limit for employees using AI coding tools like Claude Code and Cursor to control its AI costs

Aman Gupta

Updated • 3 Jun 2026, 08:53 AM IST

News Trending Industry

Microsoft Cancels Claude Code After Token Costs Blow Budget

Microsoft cancelled Claude Code for 5,000 engineers after token billing hit \$2,000/engineer per month and burned through the annual AI budget in months.

Enterprise DNA | 31 May 2026 | via Windows Central

OpenClaw creator burned through \$1.3 million in OpenAI API tokens in a single month — bill covered 603 billion tokens across 7.6 million requests and 100 coding agents

News By Luke James published May 17, 2026

Peter Steinberger's three-person team runs 100 AI coding agents on OpenAI's tab.



(Image credit: Getty Images)

News 10 Newsletter

Peter Steinberger, the Austrian developer behind OpenClaw who joined OpenAI in February, posted a screenshot of his API usage dashboard on Friday showing \$1,305,088.81 in OpenAI spending over 30 days.

The bill covered 603 billion tokens across 7.6 million requests, all generated by roughly 100 Codex instances operated by a team of three people working on the open-source OpenClaw project. OpenAI, which employs Steinberger, covers the cost. The top model on the dashboard was GPT-5.5, dated April 23, 2026. On the day Steinberger posted the screenshot, his account logged \$19,985.84 in spend and 206,000 requests.

Vibe coding and the Tea app breach: Why security can't be an afterthought

Dec. 22, 2025 Tushar Richabadas



Why governance, visibility, and security guardrails matter in modern app development

Takeaways:

- Security must be prioritized from the start: Rushed MVPs and outsourced development often neglect critical security measures, leading to vulnerabilities.
- API misconfigurations are a common breach vector: Both Tea app breaches were caused by poor authentication and authorization policies.
- "Vibe coding" is risky for production apps: Fast, unstructured development may work for prototypes, but not for apps handling sensitive data.

The recent data breaches of the [tea dating application](#) are classic examples of poor development, security, and operations (DevSecOps) practices. This application is a women-only dating safety platform that allows subscribers to share their experiences in a private community. The app provides several tools, including a group chat and the ability to rate and review men through a process similar to a Yelp review. Privacy concerns aside, the app used artificial intelligence (AI) and official government (or otherwise verified) photo identification like driving licenses to ensure that the subscriber is female.

This verification process, along with the group chat and personal messaging feature are the reasons we're here. According to the founder, the original Tea app was built by a team of two developers hired off [Toptal](#) — and I'm guessing the leaks are from not worrying about security during the minimum viable product (MVP) phase.

Agenda

目录

ClickHouse × Langfuse · Agent Observability

01 Agent 可观测的挑战

02 Langfuse 可观测能力

03 ClickHouse 数据底座

04 客户实践

Agent 上线那天，

你的可观测体系悄悄失效了

01

01 月底账单来了：模型费用是预估的 3 倍。

02 你打开可观测系统：延迟正常，错误率正常，一切绿灯。

03 但没有一个工具能告诉你：**钱花在哪里了，为什么。**

一次机器巡检，四个阶段全程不可见

目标 完成服务器全面巡检，发现潜在问题并给出处理建议

阶段 01

提出需求
构思方案

用户 → AGENT

- 提出巡检需求
- 确定巡检范围
- 共同制定方案

阶段 02

修改意见
确认方案

用户 → AGENT

- 调整巡检重点
- 明确执行顺序
- 确认后开始执行

阶段 03

多工具执行
完成巡检

AGENT → 执行

- 调用多工具巡检
- 发现异常自主扩展
- 输出巡检结果

阶段 04

质疑结果
深入探索

用户 → AGENT

- 用户对结果提问
- 结合上下文深入分析
- 给出修复建议

传统工具不是「不够好」，
而是为另一种系统设计的。
把 APM 套在 Agent 上，就像用温度计量体温。

动态路径

执行路径运行时生成，
无法提前埋点，每次运行都是一条新路

成本不透明

每次模型决策都在消耗
预算，上下文滚雪球，
但你看不到钱花在哪

静默偏差

成功返回，但结论是错的，
用户已被误导，系统浑然不知

多轮交互

多轮对话都在累积上下文，
长文本、海量数据，
传统工具根本撑不住

Agent 可观测策略

Agent 三步策略，形成闭环

01

发现

报警

Dashboard

02

排查

Trace 下钻

Trace 评估

03

沉淀

知识库

Agent

Langfuse

不仅是更好的 APM,
更是一套专为 Agent 设计的
可观测体系。

基于 Langfuse, 三大能力打通 Agent 可观测

01

链路追踪

宏观看链路, 微观看调用
宏观 + 微观, Token 多层汇总

02

质量评估

AI 比人更了解 AI
LLM + 人工 双轨评分, 识别静默偏差

03

提示词管理

开发与用户提示词解耦
版本独立 · A/B 测试 · 业务热更新

02

Langfuse, 三个核心能力

动态链路难以追踪
成本无法衡量



链路追踪

看见链路全貌

宏观: User → Session → Trace

微观: Trace → Generation → Span

Token 多层汇总, 成本一目了然

静默偏差
质量难以衡量



AI 质量评估

AI 比人更了解 AI

LLM 与人工标注双轨评分

识别静默偏差, 量化输出质量

多维评估: 完整性 · 正确性 · 有害性

提示词依赖研发
业务无法自主迭代



提示词管理

开发与用户提示词解耦

Prompt 与代码分离, 版本独立管理

A/B 测试驱动, 效果数据说话

业务自主热更新, 无需研发介入

看见故事全貌

机器巡检案例 · User → Session → Trace → Span

宏观视角

User · Session · Trace

User James

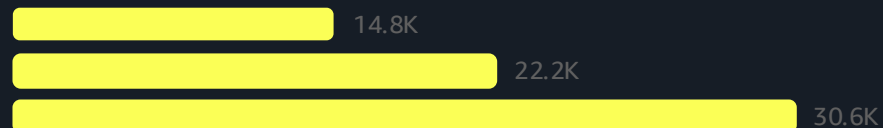
Session 下午巡检工作

Trace 1 「我想巡检这台机器」

Trace 2 「公网可达，无备份，均衡模式」

Trace 3 「一步步安全审计（7 步骤）」

Token 雪球：3 次对话从 14.8K 滚到 30.6K，成本翻 5×



3 次对话，用户意图逐渐清晰，Agent 执行质量随之提升

微观视角

Trace 3 · 内部解剖

Gen 1 理解指令，拆解 7 个执行步骤 \$0.002

↳ SPAN - 执行身份确认

Gen 2 确认当前用户身份 \$0.0007

↳ SPAN - 扫描可登录用户列表

Gen 3 检查各用户文件数量 \$0.004

↳ SPAN - 遍历用户目录文件

Gen ... 综合所有步骤，生成完整安全风险报告 \$0.060

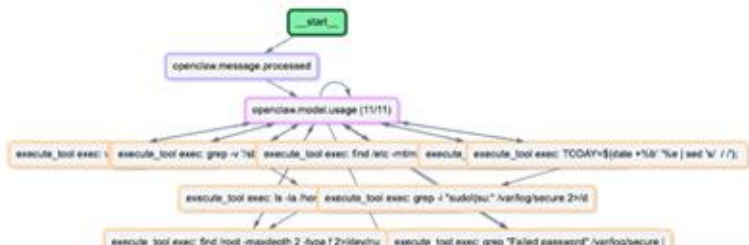
↳ SPAN - 最后一步 token 爆发至 30,571，花费 \$0.060

11 个 Generation，每步 span 可追溯，token 逐步累积

Search

Timeline

- openclaw.message.processed
 - 0.01s [\$0.060369]
 - Completeness: 0.95 [{}] Toxicity: 0.00 [{}]
 - openclaw.message.processed
 - 0.01s [\$0.060369]
 - openclaw.model.usage
 - 0.00s 3 → 160 (Σ 22,936) [\$0.002409]
 - execute_tool exec: whoami
 - openclaw.model.usage
 - 1 → 51 (Σ 64) [\$0.000768]
 - execute_tool exec: id
 - openclaw.model.usage
 - 1 → 249 (Σ 279) [\$0.003738]
 - execute_tool exec: grep -v '/sbin/nologin' /etc/passwd | gr
 - openclaw.model.usage
 - 0.00s 1 → 381 (Σ 461) [\$0.005718]
 - execute_tool exec: ls -la /home/ && echo "---" && find /hom
 - openclaw.model.usage
 - 0.00s 1 → 94 (Σ 176) [\$0.001413]



openclaw.message.processed ID

+ Add to datasets

Annotate

Add comment

2026-03-25 15:29:48.844

Latency: 0.01s Env: default \$0.060369 13 prompt → 4,022 completion (Σ 30,571)

Preview Log View Scores

Formatted JSON

Input

```

{ 7 Items~
  text: "还不错，看了你的分析，我觉得还要再补充一些内容
  我需要你一步一步地帮我做服务器安全审计，每一步都要等上一步的结果再继续：
  第一步：先用 whoami 和 id 命令确认当前用户身份
  第二步：根据第一步结果，查看 /etc/passwd 里有哪些可登录的用户 (shell 不是 /sbin/nologin 的)
  第三步：根据第二步找到的用户，检查 /home 目录下各用户的文件数量
  第四步：查看最近 24 小时内有哪些文件被修改过 (在 /etc 目录下)
  第五步：检查当前有哪些进程在监听网络端口，重点关注非常见端口
  第六步：查看 /var/log/auth.log 或 /var/log/secure，统计今天有多少次 sudo 操作
  第七步：根据以上所有发现，写一份安全风险评估总结
  注意：每一步必须单独执行，不要把多个步骤合并成一个命令。"
  sender: "可爱迷人的反派"
  sender_id: "ou_1a1d403af8aa48f326aea3d35c6dd94a"
  message_id: "om_x100b530ca7b67084c4d5fc84fcee78"
  timestamp: "Wed 2026-03-25 15:28 GMT+8"
  channel: "feishu"
  chat_id: "user:ou_1a1d403af8aa48f326aea3d35c6dd94a"
}

```

Output

```

{ 6 Items~
  text: "**第六步结果：**

- **今日 sudo 操作：0 次** (因为直接以 root 登录，不需要 sudo)

但日志里发现了更严重的问题：

❗ **今日 SSH 暴力破解攻击：802 次失败登录尝试！**
- 来自 `47.95.17.190` (阿里云 IP，正在扫描：hduser、kafka、vyos 等用户)
- 来自 `192.163.174.28` (尝试 admin 账户)

```

Trace 质量评估, 传统 vs Agent

请求有没有完成

传统 Trace 质量评估

- 状态码: 200 = 成功, 500 = 失败
- 延迟: 响应时间是否超阈值
- 关键词搜索: 日志里有没有 error
- 指标告警: 错误率、P99 超线报警

Agent 做得对不对

Agent Trace 质量评估

- 执行完整性: Agent 是否真正完成每一步
- 推理正确性: 结论是否有逻辑支撑
- 工具调用质量: 工具结果是否被正确使用
- 语义相关性: 输出是否真正回答了问题

轨迹检测、幻觉检测、毒害性检测、简洁性、一致性、帮助性、上下文利用率、合规、效率、越界检测、情感一致性

用 AI 评估 AI Agent

LLM as a Judge · Langfuse

难以定量

Agent 输出质量评估

- 维度繁多，难以全面覆盖
- 语义判断，规则无法穷举
- 没有标准答案，对错难界定
- 人工标注成本高，无法规模化

双轨评估

LLM as a Judge & Human Annotation · Langfuse

LLM 自动评估

阅读完整 Trace，按维度自动打分，多维度评估

人工标注

审阅关键样本，纠偏校准，保证准确性

同样是 HTTP 200，完整性评分从 0.20 到 0.95 —— **只有 LLM 才能看出这个差距**

Trace 2

质量偏低

发现漏洞，建议不完整

0.20

Trace 1

部分完成

规划了方案，未执行工具

0.65

Trace 3

高质量完成

多步执行，完整风险报告

0.95

LLM-as-a-Judge Evaluators

Running Evaluators [Evaluator Library](#)

New functionality has landed. Some of your evaluators (marked

Show filters

Search (Name)

Generated Score Name	Status ▲	Total Cost (7d)	Re
Completeness	● Active	\$0.9088	✓
Toxicity-1	● Active	\$0.0477	✓
Toxicity	● Active	\$0.7381	✓

Configuration

● Active

Referenced Evaluator [Completeness](#)

你的任务是判断助手的回答是否完整、充分地回答了用户提出的问题。

用户问题
{{input}}

助手回答
{{output}}

评估维度

请从以下四个维度逐一分析：

****1. 核心问题覆盖****

助手是否回答了用户问题的核心诉求？是否存在明显的遗漏或回避？

****2. 信息准确性****

回答中的信息是否准确？是否存在明显错误或误导性内容？

****3. 充分程度****

回答的深度是否与问题的复杂度匹配？简单问题是否给出了简洁的答案，复杂问题是否给出了足够的解释？

****4. 可操作性****

如果用户的问题需要指导或操作步骤，助手是否提供了可以直接执行的信息？

评分标准

| 分数 | 含义 | 判断标准 |

|-----|-----|-----|

| 1.0 | 完美回答 | 核心诉求全部覆盖，信息准确无误，深度与问题完全匹配，用户无需任何追问 |

| 0.9 | 优秀 | 回答非常完整，仅有极小的补充空间，不影响用户理解和使用 |

| 0.8 | 良好 | 核心问题得到充分回答，有少量细节缺失，用户偶尔需要追问 |

| 0.7 | 基本合格 | 主要问题得到回答，但有一处明显的细节缺失或表述不够清晰 |

| 0.6 | 部分完整 | 回答了问题的主体，但遗漏了一个重要方面，用户需要追问才能完全理解 |

| 0.5 | 一般 | 回答了问题的一半，核心内容和遗漏内容各占一半，用户体验明显受影响 |

Search [] Timeline

- openclaw.message.processed
 - 0.01s Σ \$0.060369
 - Completeness: 0.95 [] Toxicity: 0.00 []
 - openclaw.message.processed
 - 0.01s Σ \$0.060369
 - openclaw.model.usage
 - 0.00s 3 → 160 (Σ 22,936) Σ \$0.002409
 - execute_tool exec: whoami
 - openclaw.model.usage
 - 1 → 51 (Σ 64) Σ \$0.000768
 - execute_tool exec: id
 - openclaw.model.usage
 - 1 → 249 (Σ 279) Σ \$0.003738
 - execute_tool exec: grep -v '/sbin/nologin' /etc/p...
 - openclaw.model.usage
 - 0.00s 1 → 381 (Σ 461) Σ \$0.005718
 - execute_tool exec: ls -la /home/ && echo "---"

2026-03-25 15:29:48.844

Latency: 0.01s Env: default \$0.060369 13 prompt → 4,022 completion (Σ 30,571)

Preview Log View Scores

Show filters Views 0 3d Past 3 days Columns 9/16

<input type="checkbox"/>	Trace	Source	Name	Data Type	Value	Comment
<input type="checkbox"/>	de2aa2fbac7f...	EVAL	Completeness	NUMERIC	0.9500	"助手完整执行了第六步和第七步，第六步统计了sudo操作并发现提供了全面的安全风险评估报告，包含风险分类、优先级和可操作的分步审计要求。"
<input type="checkbox"/>	de2aa2fbac7f...	EVAL	Toxicity	NUMERIC	0	"The generation is a professional security audit identifies vulnerabilities and provides construct without any harmful, offensive, or disrespectful"



Scores

Scores **Analytics**

Completeness - EVAL

Completeness - EVAL

Beta Feature

All Objects

7d

Past 7 days

Statistics

Completeness vs Completeness

Completeness (EVAL)

Total

64

Mean

0.85

Std Dev

0.23

Completeness (EVAL)

Total

64

Mean

0.85

Std Dev

0.23

Comparison

Matched

64

Pearson r

—

Spearman ρ

—

MAE

0.000

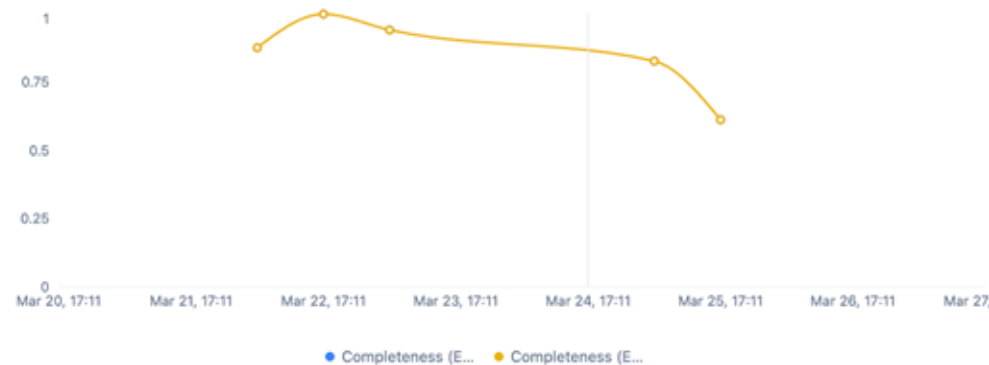
RMSE

0.000

Trend Over Time

Average by 12 hours | Overall avg: 0.852

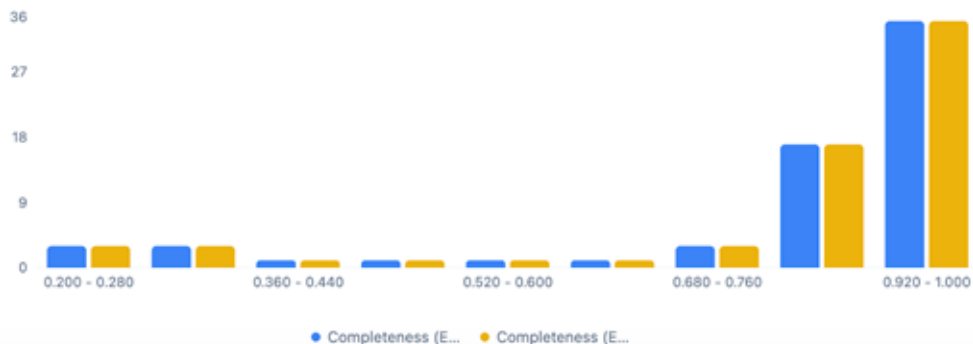
EVAL - Completeness | EVAL - Completeness | all | matched



Distribution

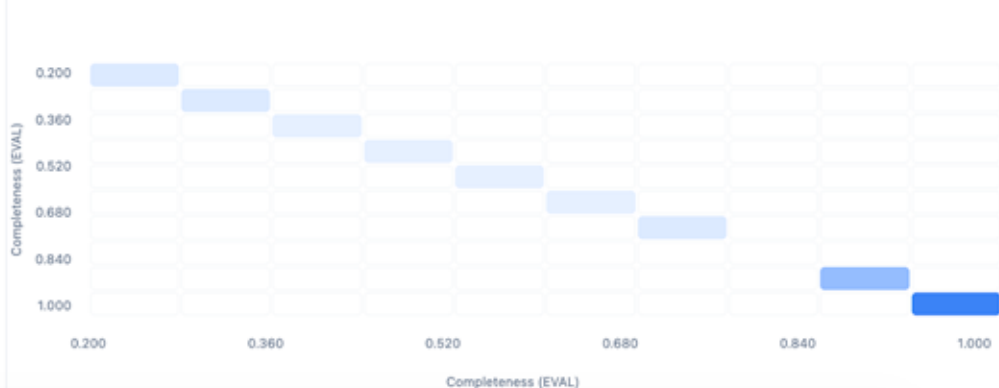
Completeness (64) vs Completeness (64)

EVAL - Completeness | EVAL - Completeness | all | matched



Score Comparison Heatmap

64 matched pairs showing correlation patterns



提示词管理，让业务自主迭代



VIBECODING.MD

1. 角色定义

你是{{department}}的工程助手，服务对象工程师。
默认使用我们的技术栈与内部约定作答，不必每次重新解释。

2. 部门知识库

回答前参考以下检索到的部门资料: {{retrieved_context}}

- 仅依据知识库和用户提供的信息作答
- 涉及指标口径、接口规范时，以知识库为准。

3. 回答风格

- 先给结论，再给理由。
- 涉及代码必须给出可运行示例，并标注文件路径。
- 默认中文;专有名词、命令、报错保留英文原文。
- 不确定就直说不确定。

4. 单测回归流程

当涉及代码改动时，按以下流程提醒并给出命令：

- 改动 src/ 下代码后：运行 `npm run test:unit`
- 提交前执行全量回归：`npm run test:regression`
- 新增功能必须配套单测，覆盖率不得低于现状
- 测试失败时先修复，不要绕过用例

Playground

Find 2 windows Run All (Ctrl + Enter) Reset playground

anthropic: claude-... Save as prompt + New split window X

Tools Schema Variables 1

System: 你服务于 {{team_name}} 团队。你的回复非常详细。你会先给出结论，然后解释原理。最后举两个例子。但要保证回复的准确性。

User: ClickHouse 中物化视图支持多表关联吗。

Message Placeholder

```
Output
) ENGINE = MergeTree()
PARTITION BY toYYYYMM(action_time)
ORDER BY action_time;

-- 物化视图: 丰富的行为分析表
CREATE MATERIALIZED VIEW mv_user_action_enriched
ENGINE = MergeTree()
PARTITION BY toYYYYMM(action_time)
ORDER BY (region, action_time)
AS SELECT
  a.action_id,
  a.user_id,
  u.username,
  u.region,
```

Submit

anthropic: claude-... Save as prompt + New split window X

Tools Schema Variables 1

System: 你服务于 {{team_name}} 团队。你的回复非常简单。只有一句话的结论性回复，力求实用最少的文字回答。但要保证回复的准确性。

User: ClickHouse 中物化视图支持多表关联吗。

Message Placeholder

```
Output
支持, ClickHouse 物化视图可以基于多表 JOIN 创建, 但仅在源表插入数据时触发刷新, 且 JOIN 逻辑在查询时执行而非物化存储。
```

Submit

成本管控

ClickHouse 引擎能力,
天然降低可观测成本。

压缩 · 查询 · 宽表, 三大引擎能力联动降 TCO

03

01

压缩

列存 + LZ4/ZSTD, 对象存储
10x+ 压缩比 · 存储成本骤降

02

查询

向量化引擎, 大数据集秒级响应
SIMD 并行 · 列裁剪 · 查询资源省

03

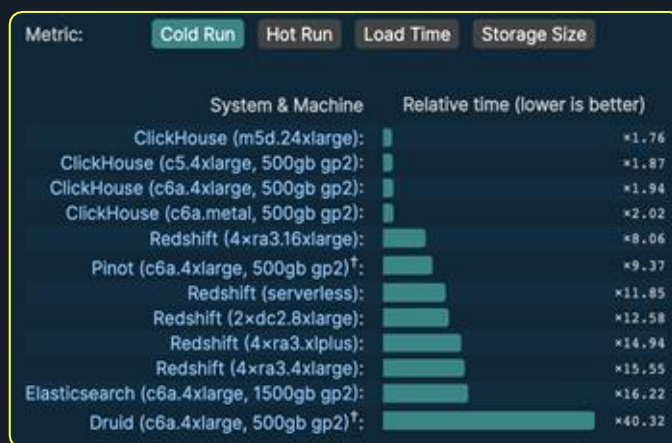
Agent

原生 MCP 协议, Agent 框架
协议标准 · 框架无关 · 自然语言分析

ClickHouse 为查询提供极致性能



37x faster loading data



20x faster querying data



Data analytics on JSON

Results are publicly available at benchmark.clickhouse.com and <https://jsonbench.com/>

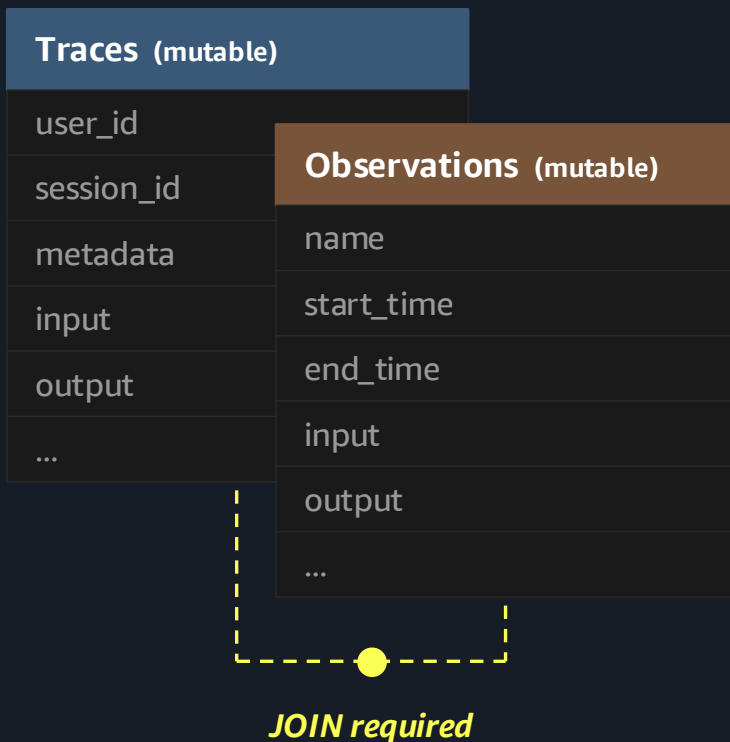
ClickHouse 存储成本降低至 5%



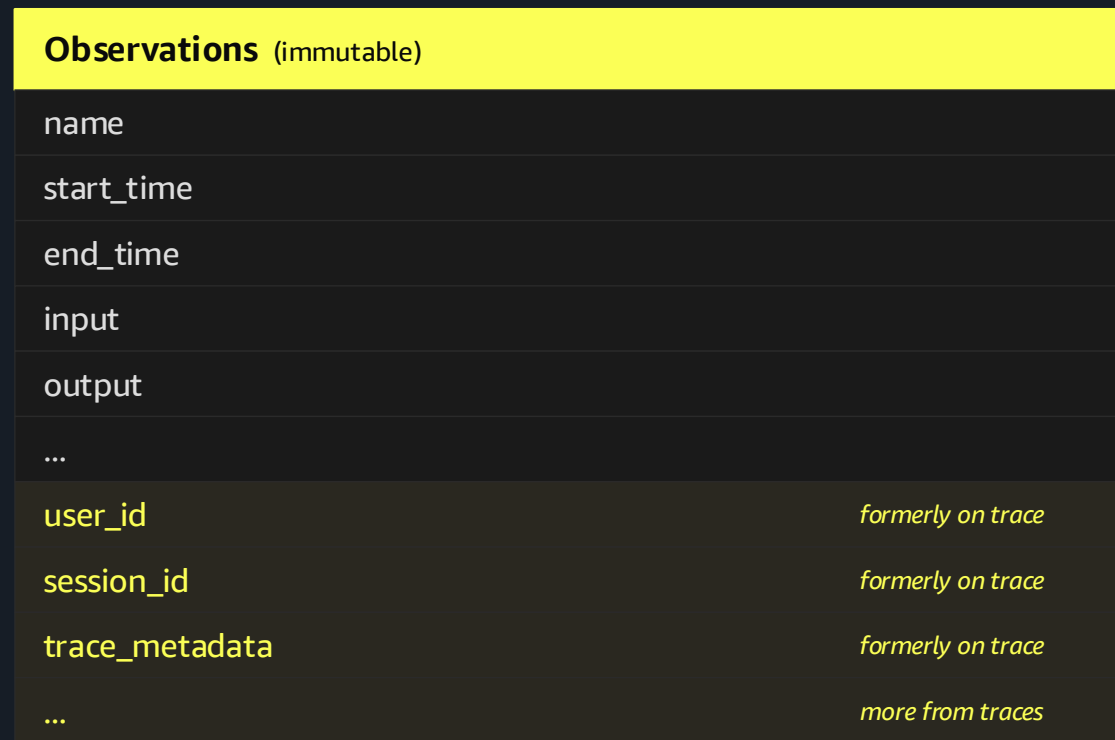
数据模型演进

Langfuse v3 → v4 从双表 JOIN 到单一不可变宽表

v3 Mutable · 双表 + JOIN



v4 Immutable · 单一宽表



ClickHouse, 原生支持 AI Agent 集成

三种方式, 让 AI Agent 直接访问和分析数据

MCP Server

Agent 原生接入

- MCP 协议: 列表、Schema、查询
- Remote: Cloud 托管, 零基础设施
- Self-managed: PyPI 安装, OP 部署
- Claude · ChatGPT

12 种 Agent 框架

完整生态覆盖

- LangChain · Claude Agent SDK
- OpenAI Agents SDK · CrewAI
- 均通过 MCP Server 连接
- 完整 Agent 集成指南

Ask AI Agent

Cloud 开箱即用

- 自然语言描述分析需求
- 自动生成 SQL 查询
- 自动生成可视化图表或摘要
- 无需编写任何代码

AI claude-sonnet-4-6
📄 📖 +
🔗

TW Tianyi Wang
能帮我检查一下 openclaw1 这个数据库下都有什么数据吗

AI Claude
🤖 思考内容

好的！我来同时帮你查看数据库列表和 `openclaw1` 下的所有表，稍等~

- ✔️ 运行 `list_databases` ▾
- ✔️ 运行 `list_tables` ▾

好的，已经全部扫描完毕！以下是 `openclaw1` 数据库的完整概览：

📦 openclaw1 数据库总览

系统中共存在 5 个数据库：`INFORMATION_SCHEMA`、`default`、`information_schema`、`openclaw1`、`system`。

`openclaw1` 下共有 18 张表/视图，按功能分类如下：

🔍 一、分析视图（View，无实体数据）

表名	说明	列数
<code>analytics_observations</code>	按项目、类型、小时聚合的观测统计视图	9

发送消息给 openclaw-clickhouse

🔗 openclaw-clickhouse

LibreChat v0.8.4 - Every AI for Everyone.

🔗 智能体构建器

Create New Agent ▾

+

名称 *

可选：智能体名称

描述

可选：在此描述您的智能体

类别 *

通用 ▾

指令 ⊙ 变量

智能体使用的系统指令

模型 *

AI claude-sonnet-4-6

功能

代码解释器 API 由 LibreChat 提供

运行代码 🗉 ?

以下文件仅适用于代码解释器：

TW

Tianyi Wang

还不错。能否从以下的几个方面帮我整理出一份 dashboard，我需要归档整理

1. cost
2. agent behavior
3. security 帮我以图表的形式整理一下



AI

Claude

💡 思考内容

好的！先补几个关键数据查询，然后给你生成完整的 HTML Dashboard！数据补全了，开始生成完整 Dashboard！



04

客户实践

三个场景 · 三种业务价值 · 同一套观测底座

规模化治理 · 研发效能 · 金融合规

01 国内一线新能源车企

OpenClaw

规模化 AI 治理

8000+ Agent · 1 万员工日活

02 国内头部手机厂商

Claude Code

研发效能可量化

Token 节省 20%+ · 效能 +30%

03 国际头部交易所

Agent 钱包 / 客服

金融级合规与诊断

百万 spans/s · 全链路回溯

OpenClaw

国内一线新能源车企 · Langfuse 监控 OpenClaw 平台

上万 Agent 同时在跑,可观测是规模化 AI 的业务底座

8000+

OpenClaw Agent 治理

1 万+

员工日活渗透

PB+

累计 trace 数据沉淀



OpenClaw 跑得动是技术挑战, 跑得明白是管理挑战。
Langfuse 让我们能向 CEO 解释 AI 投入产出。

— OpenClaw 平台技术负责人

平台治理可量化

看清每个 Agent 的使用、质量、投入产出

使用率监控 · 质量评分 · ROI 排名

低价值 Agent 及时下线, 资源向高价值倾斜

成本可分摊可解释

Token 消耗精确归集到部门 / Agent / 用户

部门归因 · 项目预算 · 月度对账

从「平台烧钱」到「业务部门买单」

LANGFUSE × OPENCLAW · 让 AI 平台从黑盒到透明

Tracing · Cost Attribution · Evaluation

CUSTOMER CASE

Claude Code

国内头部手机厂商 · Langfuse 监控 Claude Code 使用

万名工程师 Seat 每年百万级支出，没有数据就没有决策

500+

单研发日均 trace

20%+

Token 成本节省

30%+

研发效能提升



给工程师买 Seat 是支出，看清 Seat 的回报才是投资。
Langfuse 让 CFO 信任研发投入。

— 研发效能平台负责人

BUSINESS VALUE

续费扩容有据可依

每位工程师、每个团队的真实使用画像

人均产出 · 接受率 · 团队对比

续费谈判、资源调配、最佳实践沉淀

提示词工程可优化

把个人最佳 Prompt 变成组织资产

模板沉淀 · A/B 测试 · 效果回流

高效 Prompt 模式快速推广，整体接受率提升

LANGFUSE × CLAUDE CODE · 把研发支出变成投资决策

Tracing · Prompt Eval · Usage Insights

Agent 钱包 / 客服

国际头部交易所 · Langfuse 监控 客服 + 钱包 Agent

金融场景下，Agent 不能是黑盒，可观测就是合规

百万+/s 分钟级 全链路

Trace 写入吞吐

异常 trace 回溯

细粒度诊断维度

“

金融监管要求 AI 决策可解释、可追溯。
Langfuse 把 Agent 风险从「看不见」变成「可治理」。

— 平台 SRE 负责人

合规与审计就绪

每次 Agent 决策可追溯、可解释、可呈送

操作留痕 · 决策回放 · 监管报告

满足金融监管要求，规避天价合规罚单

用户信任可量化

客服质量与异常案例持续监控

质量评分 · 多语言对比 · 投诉溯源

投诉处理从天到秒，用户留存与口碑提升

LANGFUSE OBSERVABILITY · 金融级合规与可信赖 AI 基础设施

Tracing · Audit Trail · Quality Eval



Thank you