Litefuse

Agent 可观测与效果评估平台

Agent生产部署遇到的可靠性挑战

不确定性的新挑战

一个真实的例子：传统监控在Agent生产环境失效

客服 Agent 调整system prompt造成回复质量下降。

系统指标　未见异常

- 请求成功率正常
- 时延、成本无明显变化
- 没有触发异常告警

业务指标　明显下滑

- 未查询订单就直接回复
- 转人工率 12% → 31%
- 满意率 4.6 → 3.9

原因

新 prompt 只保留了语气要求，订单查询 SOP 被覆盖，Agent 不再调用查询工具。

请求链路仍然成功，因此错误率和时延监控均未发现异常。

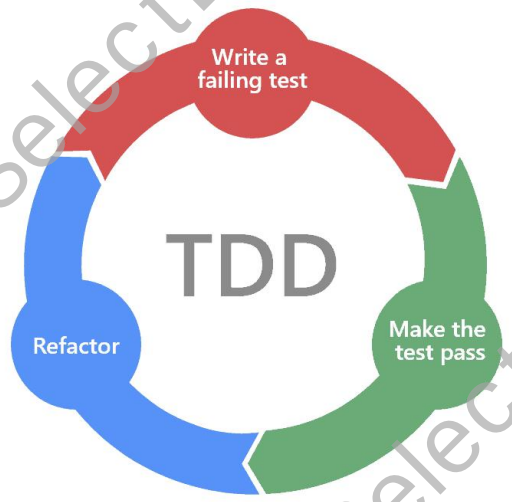
传统监控能发现服务故障，却无法判断 Agent 完成任务的质量

传统软件故障	
业务逻辑错误	峰值流量
运行时异常	基础设施故障

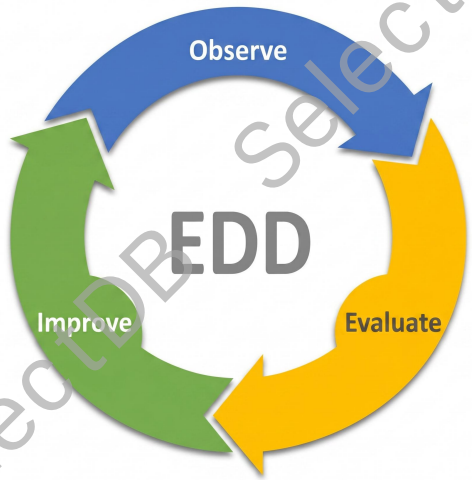
Agent 任务失败	
生成内容错误	执行路径偏离
工具调用缺失	上下文错误

TDD 验证确定性逻辑，EDD 评估 Agent 的执行过程与结果

Test-Driven Development



Evaluation-Driven Development



用 EDD 定位并防止这次 Prompt 调整造成回复质量下降

Observe 观察

- 筛选转人工会话，查看 Trace
- 对比 prompt 更新前后的执行路径
- 发现物流咨询没有查询订单

Evaluate 评估

- 从真实咨询中抽取 100 条用例
- 构建SOP执行评估器
- SOP 执行通过率作为打分指标

Improve 改进

- 回归测试通过率验证
- 加入 prompt 变更回归测试

每次修改 prompt，都用固定用例验证关键 SOP 和工具调用。

02

Litefuse 解决方案

Agent 可观测与效果评估平台

Litefuse – 将 EDD 产品化的 Agent 可观测与效果评估平台

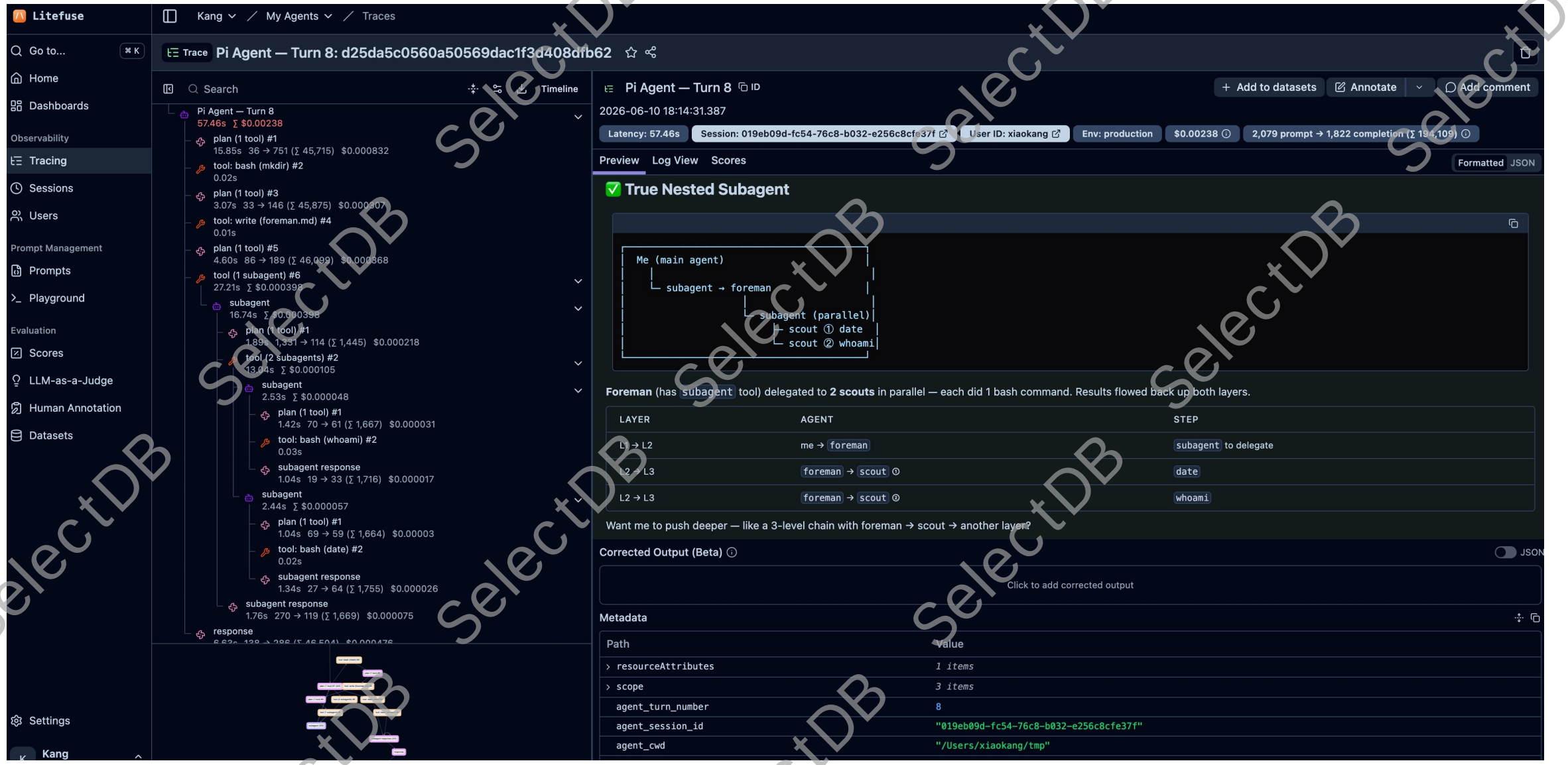
Observability

- SDK/OTel Trace 采集
- 100+ AI 生态集成
- Trace 检索与可视化
- 性能、成本等指标分析

Evaluation

- 离线测试数据集评估
- 在线 Trace 回流评估
- 人工标注评估
- LLM / 程序自动评估

Litefuse - Observability 解开 Agent 执行黑盒



Litefuse - Evaluation 评估 Agent 效果

Litefuse 基于 Langfuse 与 SelectDB 构建

两类难点，两个基石各解一类 —— Langfuse 补功能完整性，SelectDB 解性能与成本

Agent 功能难点

- AI 生态多、集成复杂
- 观测与评估割裂

Agent 数据难点

- MB 级长文本，检索慢，查询内存占用高
- 超长 Trace，获取整个 Trace 查询慢
- 大量 JSON 字段查询慢
- 数据量大、存储成本高

Langfuse 解决

- √ 100+ AI 生态开箱对接
- √ 可观测 + 评估一体化

SelectDB · Apache Doris 解决

- 倒排索引 · 文本秒级检索，延迟物化 · 按需加载内存
- 分桶排序 · Trace 聚簇连续存储，点查减少 IO
- VARIANT · JSON 列存加速和稀疏万列
- 存算分离 · 数据写入一次，对象低成本存储

Litefuse 产品优势

01

单机版极简轻量

安装包<400MB

无需Docker · 25秒安装

02

成本大幅降低

存储成本

最高降 88%

03

Agent 生态丰富

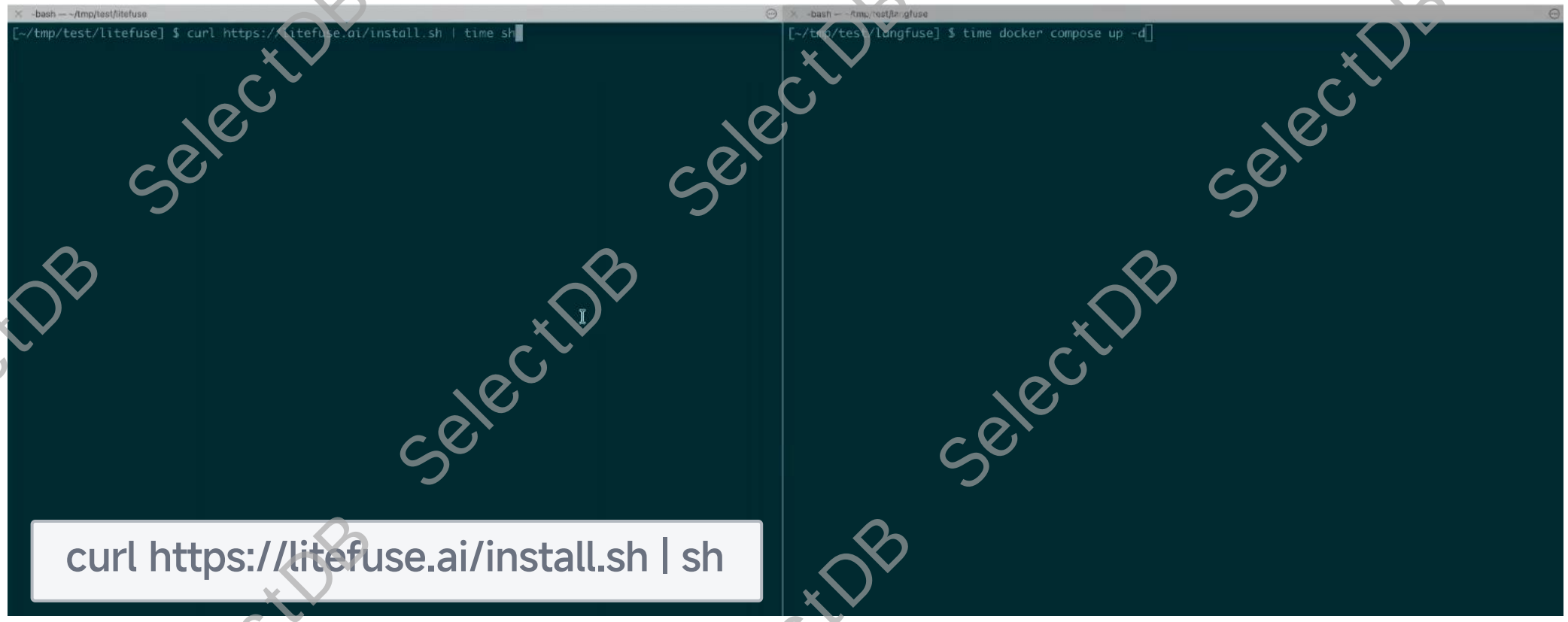
大幅增加 Agent 支持

Hermes · OpenClaw 等

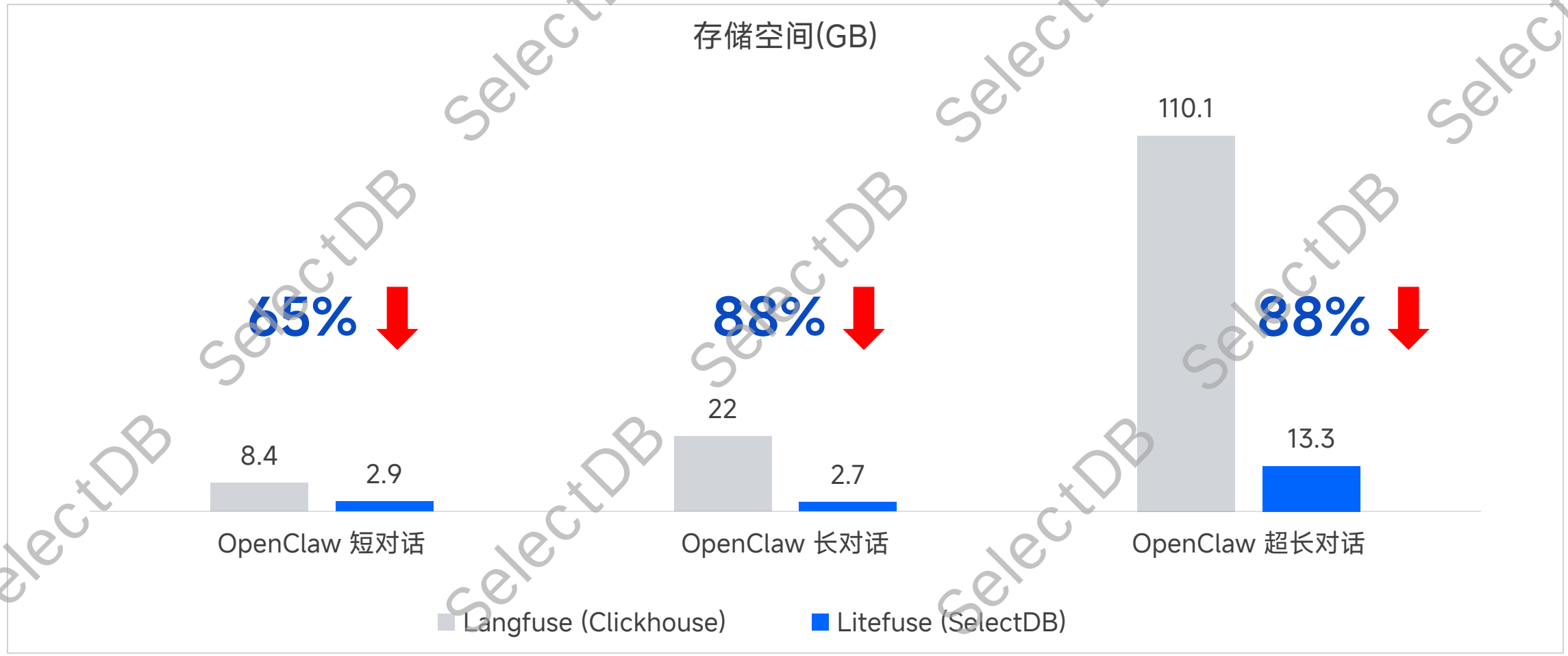
Litefuse 产品优势 – 单机极简轻量

安装包 <400MB	无需外部依赖	无需 Docker 环境	单进程运行

Litefuse Standalone · 25s VS. Langfuse Docker · 138s



Litefuse 产品优势 - 成本节省 实测最高 88%

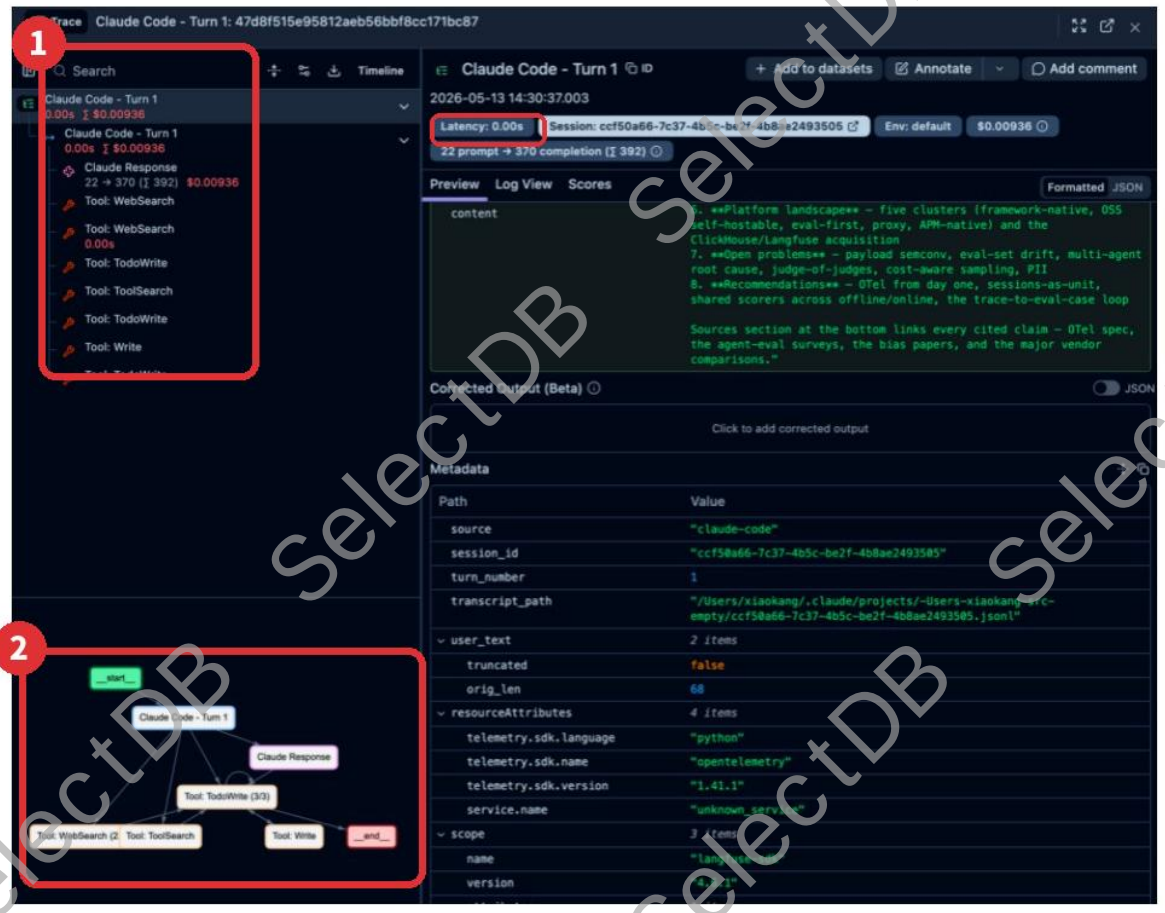


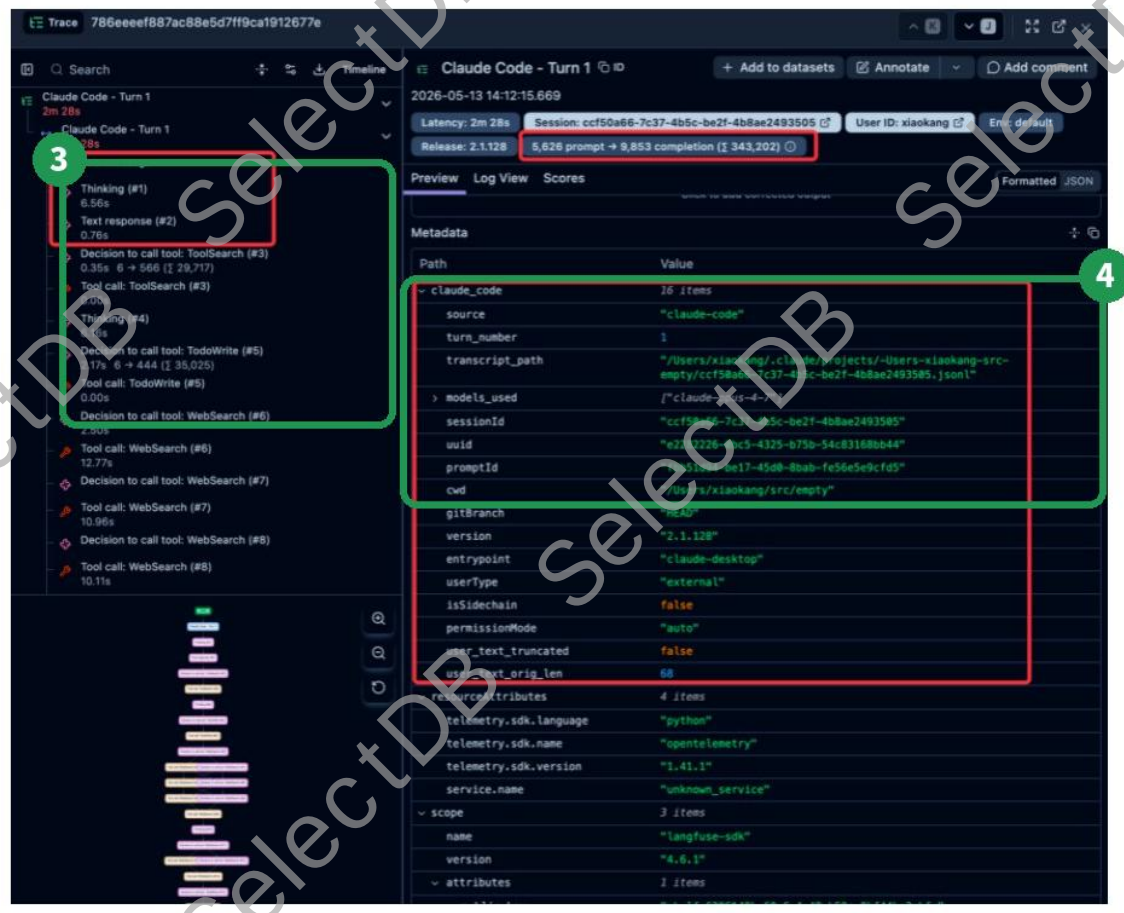
Litefuse 产品优势 - Agent 生态丰富

在 100+ AI 生态集成基础上，大幅增加 Agent 对接

Claude Code	Codex	Cursor
OpenClaw	Hermes	Pi
MiniMax Coder	Kimi Code	StepClaw

Litefuse 产品优势 - Agent 生态丰富





Langfuse — 看不全、看不准

1. 时间戳全是 0.00s，时间线完全失真
2. 执行图错乱；trace 里只有工具名，没有过程

Litefuse — 完整、准确

3. thinking → 工具决策 → 响应每一步都在，耗时真实
4. claude_code 上下文完整：session/turn/分支/版本

Litefuse 三种产品形态

Open Source

- ✓ 单机版 · 极致简洁轻量
- ✓ 分布式版 · 成本节省 50%

开源免费

Enterprise

+ 含 OpenSource 全部能力

- ✓ 深度优化 · 成本节省 80%
- ✓ 企业级特性 Coming Soon

企业自建

Cloud

+ 含 Enterprise 全部能力

- ✓ 免维护 开箱即用
- ✓ litefuse.cloud
- ✓ 阿里云 SelectDB

云上托管

从 litefuse.cloud 快速开始

10万条/月

开发者够用的免费额度

10万/30天/12小时/10 = 27 对话/小时

300+ 用户

上线一个月

真实开发者自发注册

背后的技术方案

Agent 可观测本质是一个海量数据规模下的实时数据分析问题

Agent 可观测技术挑战 – 典型场景与查询模式

场景 1 · 问题排查

“这条 Trace 从哪一步开始跑偏 / 报错？”

场景 2 · 效果评估

“换 prompt / 模型后，质量和工具选择变好？”

场景 3 · 成本分析

“哪个 model / 工具 在烧钱，token 花在哪？”

1

Trace 元数据筛选

按 model / status / cost / 时间窗过滤出目标 Trace

2

Trace 文本检索

在 input / output 长文本里按关键字和短语定位

3

Trace 点查

一次取回某条 Trace 的全部 Span 做 Agent 轨迹分析

4

Trace 统计分析

跨海量 Trace 算 token / cost / 成功率 / P99

三种典型场景归结为围绕 Agent Trace 的四种查询分析模式

Agent 可观测技术挑战 - Trace 量变引起质变

① 长文本

KB → MB

LLM 单次请求输入 / 输出达 MB 级，传统仅百字节 ~ KB

长文本检索慢，查询内存易撑爆

② 超长 Trace

MB → GB

跨度几分钟到几天，由几百 ~ 几万个 Span 组成

取全 Trace 需读大量分散文件

③ 半结构化 JSON

元数据 → 主数据

input / output 主数据全是复杂嵌套 JSON，

读取解析整段 JSON 效率极低

④ 数据量跃迁

几十 TB → 百TB

单 Span 变大 × Span 数变多，日增百 TB

存储与写入成本数量级上升

Trace 数据的量变让传统可观测技术的假设失效，无法满足需求

Agent 可观测技术挑战 - Litefuse 技术方案

Agent 可观测挑战	Litefuse · SelectDB / Apache Doris 方案	量化收益
MB 级长文本	**倒排索引全文检索 + 延迟物化降内存** 成熟倒排索引支持 MATCH / 短语 / 正则；延迟物化排序阶段只读时间字段	**Trace 全文检索 10x**
超长 Trace	**trace_id 分桶 / 排序 / 前缀索引，聚簇连续存储** 一个 Trace 聚簇落在同一节点相邻区域，范围 scan 直接取全 Trace，减少随机IO	**Trace 点查 5x**
大量嵌套 JSON	**VARIANT 类型自动拆子列、列式存储、免解析** 写入时识别字段拆子列，查询只读相关子列、无需 JSON 解析	**Trace 元数据筛选 10x**
数据量跃迁	**存算分离 (对象存储仅一份) + 列存 + ZSTD** 数据存对象存储只存 1 份而非 2~3 份，写入也只写一次	**成本 ↓75–88%**

Litefuse 技术方案 - 倒排索引加速全文检索 10x

对应挑战 MB 级长文本 —— 用户只发来一张截图、没有 Trace ID，要按关键字到 PB 级日志里反查，LIKE 越扫越慢

运行机制

长文本字段
input / output

→

倒排索引
词 → Span 列表

→

短语检索
MATCH_PHRASE

→

秒级返回
定位到 Trace Span

关键能力

- MATCH / MATCH_PHRASE / 前缀 / 词距 SLOP
- 正则 MATCH_REGEXP · 多字段 MULTI_MATCH
- 中 / 英 / IK / ICU 多语言分词
- BM25 相关性打分与排序
- SQL 与 Lucene 两种检索语法

示例 SQL（Apache Doris 语法）

```
CREATE INVERTED INDEX idx_input
  ON spans(input) PROPERTIES('parser'='unicode');

SELECT trace_id, ts FROM spans
WHERE input MATCH_PHRASE '生成失败 timeout'
ORDER BY ts DESC LIMIT 10;
```

PB 级存储 · 百 GB/s 写入 · 秒级检索
2023年起在 MiniMax、阶跃星辰、字节、快手等生产验证

Trace 全文检索提速 10x

Litefuse 技术方案 - Trace 聚簇存储 高效点查

对应挑战 超长 Trace —— 几万个 Span 散落在成百上千个文件里，复盘一条完整轨迹要到处扫文件

运行机制

trace_id hash 分桶
同 Trace 落同一节点

➔

文件内按 trace_id 排序
物理聚簇 · 数据局部性

➔

前缀索引 + 范围 scan
定位即取回 · 不跨文件

关键能力

- trace_id hash 分桶，数据局部性好
- 文件内按 trace_id 排序、物理聚簇
- trace_id 前缀索引快速定位
- 范围 scan 一次取回整条 Trace
- 复用面向用户实时分析的成熟机制

示例 SQL（Apache Doris 语法）

```
CREATE TABLE spans (trace_id STRING, ts DATETIME, ...)
DISTRIBUTED BY HASH(trace_id) BUCKETS 32
ORDER BY (trace_id, ts);


-- 取整条 Trace：一次范围 scan，不跨文件
SELECT * FROM spans WHERE trace_id='tr_8f2a…';
```

为实时分析而生的分区分桶机制
天然适配超长 Trace 读取

取全 Trace 一次 scan

Litefuse 技术方案 - VARIANT 类型加速 JSON 分析 10x

对应挑战 大量嵌套 JSON —— input / output 主数据全是复杂嵌套 JSON，读取、解析整段 JSON 效率极低

运行机制

```
{ "model": "opus-4",
  "usage":
{ "input_tokens":… },
  "tool_calls": [ … ],
  "total_cost": 0.07 }
```

➔

- 子列 model
- 子列 usage.input_tokens
- 子列 tool_calls[]
- 子列 total_cost

关键能力

- 写入时自动识别 JSON 字段名与类型
- 拆成子列、列式存储，压缩率高
- 查询只读相关子列，读取量大降
- 无需耗时的 JSON 解析过程
- input/output 收益比 metadata 更大

示例 SQL（Apache Doris 语法）

```sql
CREATE TABLE spans (input VARIANT, output VARIANT);

-- 查询只读相关子列，无需 JSON 解析
SELECT input['model'],
       input['usage']['input_tokens'],
       output['tool_calls']
FROM spans WHERE input['model'] = 'opus-4';
```

写入自动拆子列 · 高压缩
schema 漂移自适应

Trace 元数据和数据筛选 10x

Litefuse 技术方案 - 存算分离降低存储和写入成本

对应挑战 数据量跃迁 —— 单 Span 变大 × Span 变多，原始数据量与存储成本都翻了一个量级

运行机制

ClickHouse（Langfuse 开源）

本地盘 × 2~3 副本

Apache Doris 存算分离

对象存储 × 1 份

份数 ↓50% × 对象存储单价 25–50% = **总成本 ↓75–88%**

关键能力

- 对象存储只存 1 份，告别多副本
- 列存 + ZSTD 压缩降空间
- 对象存储单价仅本地盘 25~50%
- 写入只写一次，省计算与 IO
- 存储、计算独立弹性扩展

示例 SQL（Apache Doris 语法）

```
-- 存算分离：数据只存对象存储 1 份（非 2~3 副本）
CREATE STORAGE VAULT s3_vault TYPE = S3 (...);


-- 列存 + ZSTD 压缩，进一步降空间
ALTER TABLE spans SET('compression'='zstd');
```

SelectDB 和 Apache Doris 均支持存算分离模式

总成本 ↓75–88%

基于 Litefuse 构建可靠的 Agent

https://litefuse.ai