9.29 从数据入手设计aiops agent

输入

Metrics

监控指标，如 CPU 使用率、磁盘 I/O 等，这类指标以定量和时序的方式反映系统健康状况，但缺乏背景信息以识别根本原因。

Logs

服务日志，组件产生的结构化或半结构化日志，日志通常包含明确的故障线索，例如“连接被拒绝”或“超时超过”等错误或警告消息，这些对诊断非常有帮助。然而，绝大多数日志条目是例行的调试或信息级别的消息，与事件无关（例如，“severity：info，message：conversion request successful”）。

Traces

调用链数据，服务之间的依赖与请求路径，调用链数据捕捉到微服务之间的详细执行路径，并可以揭示依赖关系异常。然而，大多数追踪记录对应于正常操作，减弱了它们在根因分析中的有效性。

Metrics、Logs、Traces 反映了系统行为的不同方面，所有这些都提供了有价值的诊断信号。然而，这些数据往往被冗余条目所主导，这妨碍了有效的分析。

输出

AD (Anomaly Detection)

异常检测，有或没有，二分类。

FT (Failure Triage)

故障类型，多分类。

RCL (Root Cause Localization)

定位导致异常发生的根本原因组件，可以是一个或多个，如"checkout-service"。

设计原则

数据精炼

原始数据包含过多无关信息妨碍了 agents 的有效分析，并且考虑到 LLM 上下文窗口大小的限制，对原始数据进行提炼操作是必要的。

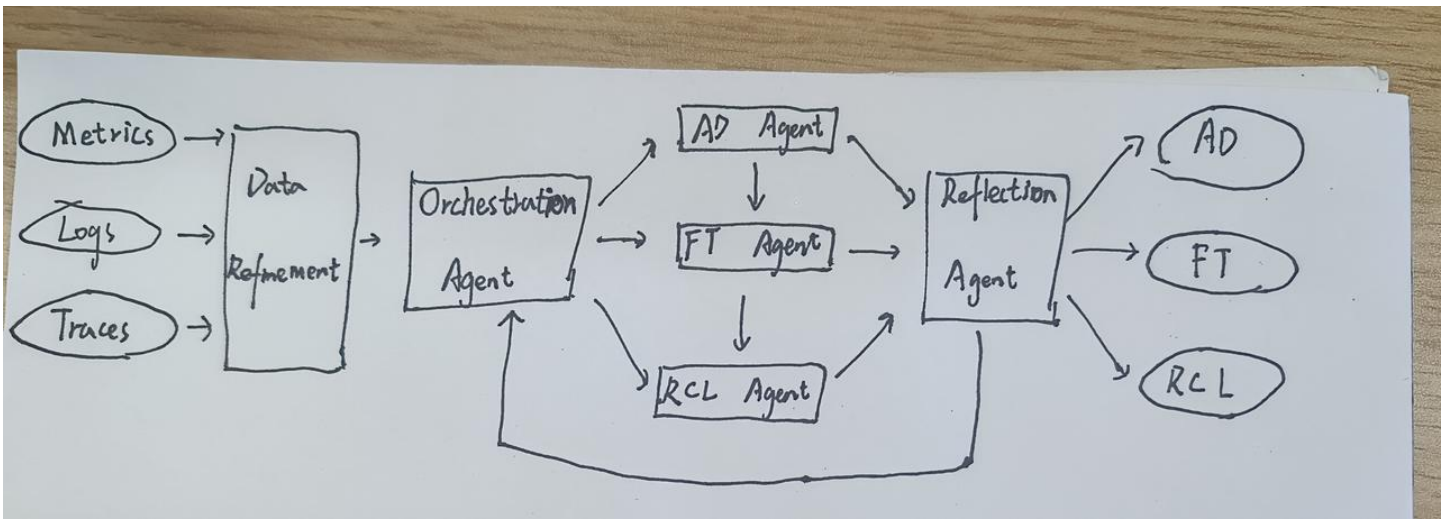
MoE (Mixture-of-Experts)

单个 agent 能力有限，通过多个 agents 组合来提高模型性能。每个 agent 负责一个子功能，最后综合多个agents 的输出得到最终输出。

反思机制

agent 的输出不一定正确，需要一种反思机制让 agent 反思自己的输出，保证结果可靠。

粗设计



Data Refinement（提取有用信息）

对原始输入数据进行提炼操作。一种简单方法是：

- 对于 Metrics，利用统计方法找出其中异常的数据，并标记异常数据出现时的时间戳，如找出 cpu 占用率过高/过低的部分。
- 对于 logs，使用关键字（warning、error等）提取其中关键的logs，并标记时间戳。
- 对于 Traces，利用上述时间戳找到对应的 traces。

Orchestration Agent

接受提炼之后的数据，并根据这些数据制定合理的分析策略，使用子Agent进行分析。

1.首先调用 AD Agent 检测是否有异常发生。

2.若发生异常则调用 FT Agent 进行故障分类。

3.最后调用 RCL Agent 进行根因定位。

4.根据 Reflection Agent 的反馈判断是否重复上述步骤。

AD Agent

异常检测 Agent，检测是否发生异常，并给出合理解释说明。将检测结果及解释说明传递给 FT Agent 与 Reflection Agent。

FT Agent

故障分类 Agent，若有异常发生时，则将故障分类，并给出合理解释说明。将分类结果与解释说明传递给 RCL Agent 与 Reflection Agent。

RCL Agent

根因定位 Agent，根据故障类别进行根因定位，找出一个或多个根因组件并给出合理的解释说明。将定位结果与解释说明传递给 Reflection Agent。

Reflection Agent

反思 Agent，接受上述三个 Agent 的输出结果，判断其结果是否一致，解释是否合理。若不一致/不合理，则反馈到 Orchestration Agent 重复任务；若合理，则输出结果。