Alibaba Agent Infra Interview

阿里 Agent Infra 工程师面试准备

AI Infra · Agent Runtime · RAG / Memory · LLM Serving · Reliability Engineering

核心定位：把 Agent / LLM 应用做成稳定、可观测、可扩展、可恢复的平台基础设施。

1. 岗位画像

方向	重点
Agent Infra	Agent runtime、工具调用、任务编排、记忆系统、RAG、评测、权限、沙箱、可观测性。
稳定性工程	高可用、超时、重试、幂等、限流、熔断、降级、监控告警、SLO、成本控制。

一句话表达：我关注如何把 Agent 从 demo 做成平台能力：任务可编排、状态可恢复、工具调用可审计、检索链路可观测、失败可降级、成本可控制。

2. 六块准备重点

模块	必须掌握
Agent 系统设计	Planner、Executor、Tool、Memory、Evaluator、Workflow 如何拆。
RAG / Memory	chunk、embedding、ANN、metadata filter、rerank、cache、权限隔离。
LLM Serving	prefill、decode、KV cache、batching、streaming、vLLM。
稳定性工程	timeout、retry、idempotency、限流、熔断、降级、SLO。
分布式系统	task queue、worker crash、checkpoint、workflow、Raft 概念。
工程基础	Python/Go/Java、Redis、MySQL、MQ、Docker、K8s、Linux。

3. Agent 平台架构

```
User Request
  -> API Gateway
  -> Planner
  -> Task Scheduler
  -> Agent Runtime
  -> Tool Executor
  -> Memory / RAG
  -> Evaluator
  -> Response Aggregator
```

- API Gateway：鉴权、限流、租户隔离。
- Planner：把用户目标拆成步骤。
- Scheduler：调度任务到 worker。
- Agent Runtime：执行 Agent loop，管理上下文。
- Tool Executor：工具调用、参数校验、沙箱、超时。

- Memory / RAG：检索知识和历史状态。
- Evaluator：质量评估、安全检查、终止判断。

4. Tool Calling 稳定性

- 工具注册中心：schema、权限、timeout、retry 策略。
- 参数校验：JSON Schema / Pydantic。
- 权限控制：不同用户、租户、Agent 可用工具不同。
- 超时控制：每个工具必须有 timeout。
- 幂等设计：可重试工具必须有 request id。
- 审计日志：记录谁在何时调用了什么工具。
- 沙箱隔离：代码执行、文件操作、浏览器操作不能裸跑。
- 结果截断：防止工具返回过大内容撑爆上下文。

5. RAG 与 Memory

```
Offline: document -> parse -> chunk -> embedding -> vector index
Online: query -> query embedding -> ANN search -> metadata filter -> rerank -> context packing -> LLM
```

Retrieval latency 拆解：

```
embedding latency
+ vector DB network latency
+ ANN search latency
+ metadata filter latency
+ fetch chunk latency
+ rerank latency
+ prompt assembly latency
```

Memory 设计字段：id、user_id、content、embedding、importance、timestamp、source、expires_at。难点是错误记忆污染、过期策略、去重合并、权限隔离和延迟控制。

6. LLM Serving

- Prefill：处理 prompt，建立 KV cache。
- Decode：一次生成一个 token，逐步追加 KV cache。
- LLM 推理慢：权重大、decode 自回归、KV cache 占显存、小 batch GPU 利用率低。
- vLLM：PagedAttention + continuous batching，降低 KV cache 碎片，提高并发。
- Streaming：优化体感延迟和 time to first token，不减少总计算。

7. 稳定性工程

机制	要点
Timeout	LLM、embedding、vector DB、tool、browser/code execution 都要有 timeout。
Retry	指数退避；副作用操作不能盲目重试。
Idempotency	request id、幂等 key、去重表、状态机。
Rate Limit	保护系统不被流量打爆。
Circuit Breaker	依赖持续失败时暂时停止调用。
Fallback	rerank 挂了退化为 vector top-k，高级模型限流切便宜模型。
SLO	Availability、P95/P99、TTFT、task success rate、tool success rate、cost per task。

8. DeepScientist 项目包装

一句话：DeepScientist 是一个面向科研任务的 multi-agent workflow 系统，本质是 mini distributed system。它把复杂科研问题拆成检索、阅读、总结、写作、评估等可恢复步骤，并通过 RAG 和 memory 管理中间知识。

```
User Query
  -> Planner
  -> Search Workers
  -> PDF / Web Parser Workers
  -> Memory / Vector Store
  -> Draft Writer
  -> Critic / Evaluator
  -> Final Aggregator
```

稳定性点：tool timeout、search/parser retry、LLM fallback、中间结果 checkpoint、trace 记录耗时、缓存搜索和 embedding、外部 API 限流。

9. 高频系统设计题

1. 设计一个企业内部 Agent 平台：多租户鉴权、tool registry、workflow runtime、memory/RAG、sandbox、observability、evaluation、model router、cost control。
2. Agent 调工具超时怎么办：timeout、按工具类型 retry、幂等工具可重试、副作用工具不自动重试、记录 trace、返回降级结果。
3. RAG 检索慢怎么排查：embedding、vector search、metadata filter、fetch chunk、rerank、context packing 分段看 P99。
4. Agent 死循环怎么办：最大 step、最大 token budget、重复 action 检测、evaluator、重新规划、人工接管。
5. 如何做 Agent 可观测性：request id、model、prompt hash、token usage、tool input/output、retrieval top-k、rerank scores、step latency、error stack。

10. 高频八股清单

方向	问题
AI Infra	prefill/decode、KV cache、vLLM、streaming、RAG vs fine-tuning、rerank、embedding cache。
稳定性	timeout/retry、幂等、熔断限流、SLO/SLA/error budget、P99 latency、服务雪崩、灰度发布。
分布式	MapReduce、Raft majority、leader 挂了怎么办、task queue 不丢任务、worker crash 恢复。
数据库/RAG	B+ Tree vs LSM、ANN、HNSW、metadata filter 顺序、cache invalidation、LLM memory。

11. 7 天冲刺计划

天数	主题	输出
Day 1	Agent Infra 架构	企业 Agent 平台系统设计图
Day 2	RAG / Vector DB / Memory	RAG 优化 checklist
Day 3	LLM Serving	一页 vLLM / KV cache 笔记
Day 4	稳定性工程	工具调用超时处理方案
Day 5	分布式系统	DeepScientist = mini distributed system 讲稿
Day 6	项目包装	3 分钟介绍 + 10 分钟深挖
Day 7	模拟面试	3 个系统设计题 + 20 个八股题

12. 自我介绍模板

面试官您好，我是古恩豪。我主要关注 AI Infra 和 Agent 系统工程，最近系统学习了深度学习框架、GPU 推理加速、分布式系统和数据库/RAG。项目上我做过一个 DeepScientist，多 Agent 科研工作流系统，里面涉及任务拆解、工具调用、RAG 检索、memory 管理、失败重试和中间状态持久化。我对如何把 Agent 从 demo 做成稳定、可观测、可恢复的平台能力比较感兴趣，也希望在 Agent Infra 和稳定性工程方向深入实践。

13. 反问问题

1. 团队现在的 Agent Infra 更偏 runtime、tool platform，还是 serving / evaluation?
2. 稳定性工程在 Agent 场景里最关注哪些指标?
3. 目前 Agent 平台最大的挑战是 tool reliability、latency、cost，还是 observability?
4. 实习生进去后会参与平台建设、稳定性治理，还是具体业务 Agent 落地?
5. 团队技术栈主要是 Java、Go、Python，还是混合?

14. 最后总结

Agent Infra 工程师不是单纯的大模型应用开发，而是 AI 时代的平台工程岗位。核心能力是把 Agent 应用做成可靠系统：可编排、可恢复、可观测、可扩展、可降级、可控成本。

```
Agent Runtime
RAG / Memory
LLM Serving
Reliability Engineering
```