

大模型助力智能运维



优维科技（深圳）有限公司



目录

01

产品体系

02

AI场景

03

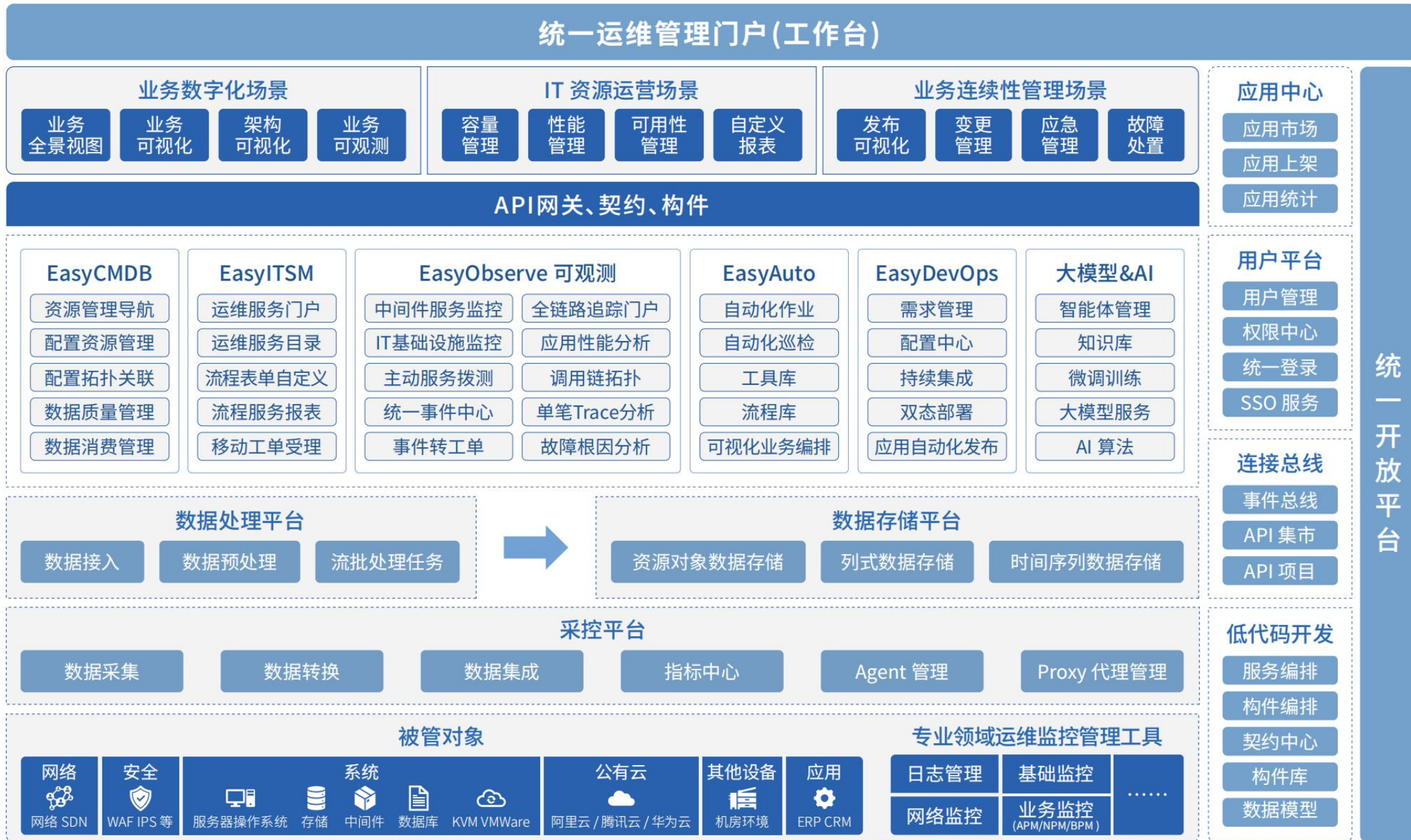
AI开发



1

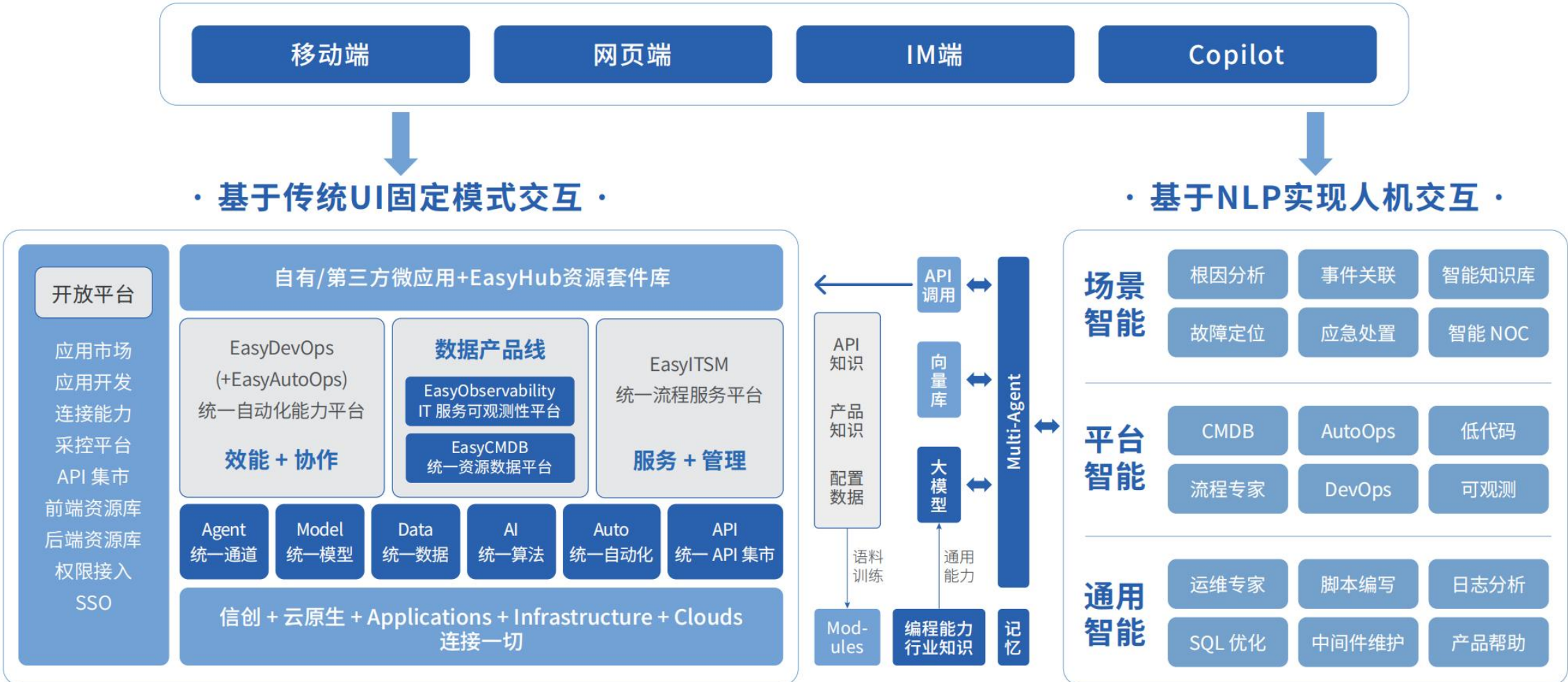
产品体系

构建新一代运维核心系统





Murphy和运维平台的人机协同



- 提供多渠道接入大模型能力
- 大模型运维能力 Copilot 化（NLP 式交互）
- 人机协同是传统运维平台与大模型的协同，而非能力孤岛
- 传统运维平台能力需要大模型对它智能化升级

- 平台提供多样数据给到向量库
- 大模型封装多能智能体，链接平台能力
- 原功能可以快速利用大模型能力升级
- 运维能力 Copilot 化，全面智能助理



2

AI场景



辅佐人，提升个人工作效率

- 百事通，知识检索，服务目录检索
- 工单助手，辅助提单，处理工单
- 告警助手，辅助故障定位，告警总结
- shell助手，辅助编写shell命令
- 编程助手，辅助编写代码，单元测试，code review
- 测试助手，辅助编写测试用例，执行用例
- 产品助手，辅助编写产品文档，出产品试题

替代人，提升企业级生产力

- NOC值班机器人
- 智能客服机器人



降本增效



3月15日 15:34





@Murphy agent状态异常怎么排查

Murphy 机器人 3月15日 15:35


针对agent状态异常的排查，可以按照以下步骤进行：

- 确认Agent状态和日志：**
 - 检查agent的easyops.log日志文件，确认是否有最新的日志记录以及是否存在错误日志。
 - 如果日志显示网络存在问题，需要检查网络设置和防火墙配置，确保agent能够正常连接到receiver服务。
- 检查Gateway日志：**
 - 查看与对应主机IP相关的gateway日志，确认是否有异常记录，如连接拒绝或超时等。
- Redis中Key的检查：**
 - 登录到Redis数据库，检查是否存在与主机IP和UUID对应的key，确认数据是否正常写入。
- 检查Easy_TSDB中的状态数据：**
 - 使用提供的命令在easy_tsdb中查询状态数据，确认状态数据是否显示异常。
 - 如果状态与实际不符，需要检查flink组件是否正常工作。
- Flink组件检查：**
 - 访问Flink UI（通常在JobManager所在服务器的8880端口），检查MetricMonitor的job是否运行正常。
 - 如果无法访问UI，或者UI显示异常，检查后台flink的日志，特别是flink_taskmanager的monitor.log，确认是否有新日志写入且日志内容正常。
 - 如果发现异常，按照顺序重启flink_jobmanager、flink_taskmanager和flink_monitor组件。

2024-05-27 13:36:54 我

 cmdb实例搜索的API接口是什么，用python写个demo 

API助手 2024-05-27 13:36:31

 CMDB实例搜索的API接口是一个POST请求，请求路由为/object/:objectId/instance/_search，服务名为logic.cmdb.service。以下是一个Python的请求示例：

```
import requests
import json

# 假设你已经有了这些参数的值
object_id = 'your_object_id' # 模型对象ID
query = {'name': {'$like': '%your_search_query%'}} # 查询条件
fields = {'instanceId': True, 'name': True, 'owner.name': True} # 需要返回的字段
only_my_instance = False # 是否只搜索与你相关实例
page = 1 # 页码
page_size = 10 # 页大小
sort = {'instanceId': -1} # 排序规则
permission = ['read'] # 权限过滤
headers = {
    'org': 'your_org',
    'user': 'your_user',
    'content-type': 'application/json'
}

# 构造请求参数
params = {
    'objectId': object_id,
    'query': query,
    'fields': fields,
```

提交



2024-04-20 16:06:35 我

今晚20点可以发布cmdb_service吗

AI助手 2024-04-20 16:06:46

部署变更影响分析报告

部署变更内容

计划在今晚20点发布cmdb_service组件。

直接影响的组件列表

["cmdb_service"]

间接影响的组件列表

[]

影响的系统列表

["EasyCMDB"]

部署规范分析过程和结果

1. cmdb_service在20点发布，这在20:00-24:00的时间窗口内，符合发布时间窗口的规定。

2. cmdb_service并未在需要使用ucpro方式部署的组件清单中，因此使用双态部署方式符合规范。

根据上述分析，cmdb_service的部署计划符合发布的规范和时间窗口要求。

部署影响分析结论

在当前时间2024-04-20 16:06:39，计划在今晚20点发布cmdb_service是可行的，符合发布的规范和时间窗口。因此，cmdb_service可以按计划进行部署。



AI场景 - 工具助手



工具库 / 新建

新建

基本信息

输入定义

输出定义

配置信息

执行时间窗口

工具是什么?

工具是一个可以在目标机器上执行并完成特定任务的脚本代码。

工具名称

分类

默认

以"/"分割分类层级

脚本语言

Shell

Python

PowerShell

Bat

脚本代码

1

说明:
使用 'InputName' 来使用输入参数, 如:
echo \$InputA

使用 'PutStr "outputName" "outputValue"' 来给输出参数赋值, 如:
PutStr "outputB" "hello"

使用 'PutRow "tableName" "rowValue"' 实现工具的表格化输出, 如:
PutRow "t1" "[ip=127.0.0.1,hostname=CentOS-1&cpu=intel-17&memory=18G"
[-----]
[ip|hostname|cpu|memory|
127.0.0.2|CentOS-2|intel-15|18G]

更多函数详情, 请查看说明中的帮助文档

调试

AI助手

分析作用

分析问题

2024-05-27 17:29:33 我

写一个磁盘清理的工具, 可指定文件路径, 指定文件修改时间

AI助手 2024-05-27 17:30:14

(1) 需求理解: 该工具用于清理指定路径下, 按照给定时间修改的特定类型的文件。用户可配置文件类型和

工具库 / AI助手

AI助手

新建会话

装载工具库

> 欢迎使用AI工具执行助手, 请向我提出需求, 我将帮你选择工具和执行

> 清理/tmp*.log下, 修改时间为2天前的文件

> 找到如下工具, 请选择:
1. 工具: 清理磁盘工具

> 你选择了【工具: 清理磁盘工具】, 它有如下参数: Execution target、路径。请输入:

> 执行参数

执行用户: root

Execution target:

指定实例

动态过滤器

快速查询策略

172.30.0.134 + 从CMDB中筛选

路径: /tmp/*.log

> 分批策略

开始执行

> 执行结果如下: [点击查看完整记录](#)

根据ip搜索

执行中(0) 成功(1) 失败(0)

172.30.0.134 成功

显示为主机名 复制IP列表

执行日志:

/tmp/*.log

回到结尾

回到开头



AI场景 – AI自动生成测试用例



⚙️

📄

📅

测试 / 功能测试 / 测试用例 / 用例详情 / 用例编辑

编辑用例

基本信息

场景名称

【CMDB_INSTANCE-2215】逻辑架构绘制能力

ID

13036

生成模式

手工

AI

AI分享链接

https://admin.easyops.local/next/ai-center/share?agentId=test-agent&robotId=&conversationId=7acfb65-e919-11ef-b234-00163e513a6d

将AI助手输出的AI分享链接粘贴到当前输入框

data

编辑 预览

以下是从给定的验收标准中设计的20条测试用例，使用了given-when-then格式，并增加了场景概述列，用例给予序号和优先级。

| 序号 | 优先级 | 场景概述 | Given | When | Then |

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

重放上下文

AI助手 2025-02-12 18:08:52

好的，以下是针对您提供的功能描述设计的20条测试用例，使用了具体的测试数据，并以given-when-then格式呈现：

序号	优先级	场景概述	Given	When	Then
1	PO	拖拽自研应用到图中	用户在左侧操作栏中选择“自研应用”	用户将“自研应用”拖拽到图中	图中出现一个自研应用的图标，图标名称为“自研应用1”
2	PO	拖拽外部系统到图中	用户在左侧操作栏中选择“外部系统”	用户将“外部系统”拖拽到图中	图中出现一个外部系统的图标，图标名称为

提出你的想法...

AI场景 – AI自动做code review



internal/app/processor/tracelog_to_sw.go 100755 → 100644

```
125 128      Key:  "kafka.message.consume.time",
126 129      Value: fmt.Sprintf("%d", msgConsumeTime.UnixMilli()),
127 130  })
131 +      span.Tags = append(span.Tags, &skpb.KeyStringValuePair{
132 +          Key:  "kafka.offset",
133 +          Value: fmt.Sprintf("%d", msgOffset),
134 +      })
128 135  }
129 136  }
130 137  }
131 138  }
132 - // createTraceLogOption 创建trace log选项参数
139 + // createHeSeeOption 创建trace log选项参数
```



Eve @eve · 1 month ago

Developer



函数注释从createTraceLogOption改为createHeSeeOption，但函数名仍然是createTraceLogOption，注释和函数名不一致，应该保持一致。

```
85 +      ConnMaxLifetime: time.Duration(10) * time.Minute,
86 +      ConnOpenStrategy: clickhouse.ConnOpenInOrder,
87 +      BlockBufferSize: 10,
88 +  })
89 +  if err != nil {
90 +      return nil, fmt.Errorf("根据连接池配置创建一个新的连接池错误: %w", err)
91 +  }
92 +
93 +  // 测试连接
94 +  if err := connPool.Ping(context.Background()); err != nil {
```



Eve @eve · 1 month ago

Developer



Ping操作应该使用带超时的context，而不是background context，避免在网络问题时一直阻塞

▼ Collapse replies

cmd/app/main.go 100755 → 100644

```
87 92      http.Handle(metricsUri, promhttp.Handler())
88 93      log.Logger.Info("定义监控指标uri完成", zap.String("uri", metricsUri))
89 94
95 +      rand.Seed(time.Now().Unix())
```



Eve @eve · 1 month ago

Developer

建议使用 `rand.NewSource()` 替代 `rand.Seed()`。从 Go 1.20 开始，`rand.Seed()` 已被废弃。可以改为：

```
rand.New(rand.NewSource(time.Now().UnixNano()))
```

或者直接使用 `math/rand/v2` 包，它不需要手动设置种子。

```
306 + // appendSpanObjectTags 添加spanObject tags指标信息
307 + func (hs *HsLogMsgProtoJson) appendSpanObjectTags(spanObject *skywalkingpb.SpanObject) {
308 +     if spanObject.IsError == false {
309 +         hs.setSpanNormalHsErrorInfo()
310 +     } else {
311 +         hs.setSpanAbNormalHsErrorInfo()
312 +     }
313 +
314 +     // 解析rep应答数据
315 +     parseKeys := make([]string, 0)
316 +     for k, _ := range hs.Option.ParsePackage {
```



Eve @eve · 1 month ago

Developer



range遍历map时使用了空标识符(_)忽略value,建议使用 `k := range hs.Option.ParsePackage` 的方式,更加简洁。



1. 给AI助手发送告警信息
2. 告警助手自动进行排查分析

The screenshot shows a chat window with a header bar containing a blue icon, the text "运维AI助理", and a "机器人" (Robot) label. The chat history includes two JSON objects: `{\"title\": \"识别意图\"}` and `{\"title\": \"生成排查计划\"}`. The current message from the user is a blue bubble containing an alert:
@运维AI助理 请帮忙分析如下告警并给出建议: 【百丽告警】2024-12-16 14:01告警
告警等级: 严重
告警资源:bjm6v-retail-tax-openapi-prd-10-250-17-58-vm.belle.lan(10.250.17.58)
告警指标:实际内存使用百分比
关联应用: retail-hx-tax-openapi
所属应用运维负责人: 刘永强,张智平
所属应用开发负责人:李凯鹏
关联交换机:10.250.255.24
告警信息:实际内存使用百分比【95.54%】,大于阈值【85%】
8人未读

The AI assistant's response is in a white bubble and includes the text "通过执行free -h命令得到了以下结果:" followed by a table of memory usage statistics:

	total	used	free
Mem:	7.6G	6.7G	143M
Swap:	0B	0B	0B

Below the table, the assistant states: "从结果可以看出,系统内存使用情况较为紧张,已用内存为6.7G,可用内存仅为317M。" and then "通过执行top -b -n 1 -o %MEM | head -n 15命令得到了以下结果:".



从系统日志找到一个java进程OOM

通过执行`top -b -n 1 -o %MEM | head -n 15`命令得到了以下结果：

```
top - 18:16:25 up 989 days, 18:35, 1 user,
Tasks: 121 total, 1 running, 120 sleeping,
%Cpu(s):  8.1 us,  3.2 sy,  0.0 ni, 88.7 id,
KiB Mem : 8008704 total, 148712 free, 702
KiB Swap:      0 total,      0 free,
```

PID	USER	PR	NI	VIRT	RES	SHR
7226	app	20	0	8489300	5.3g	2204
1103	root	20	0	2603124	933720	1612
20876	root	20	0	803312	164976	1952
13314	root	20	0	1431040	57280	4384
2404	root	20	0	1323092	36788	6564
13395	root	20	0	745852	27564	1712
19249	root	20	0	996204	22656	3660
13394	root	20	0	311116	18504	2392

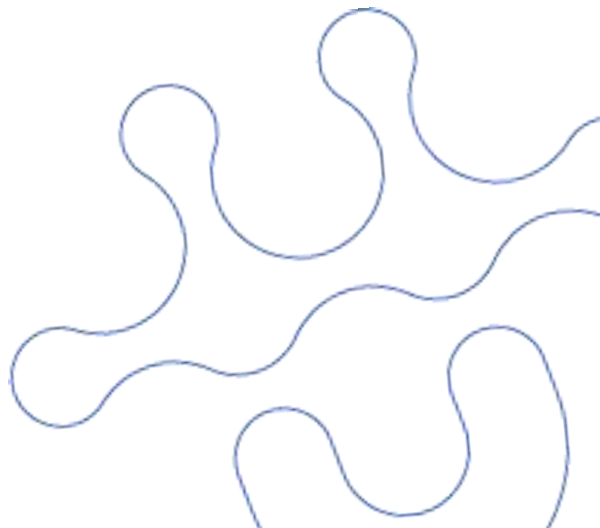
从结果可以看出，PID为7226的java进程占用了大量内存，内存使用率为69.2%。

通过执行`dmesg -T | egrep -i 'killed process' | tail -5`命令得到了以下结果：

```
[Sat May 25 04:26:58 2024] Killed process 266
[Fri Aug 2 04:26:32 2024] Killed process 235
[Sat Sep 28 11:49:35 2024] Killed process 189
[Fri Oct 25 20:33:56 2024] Killed process 887
```

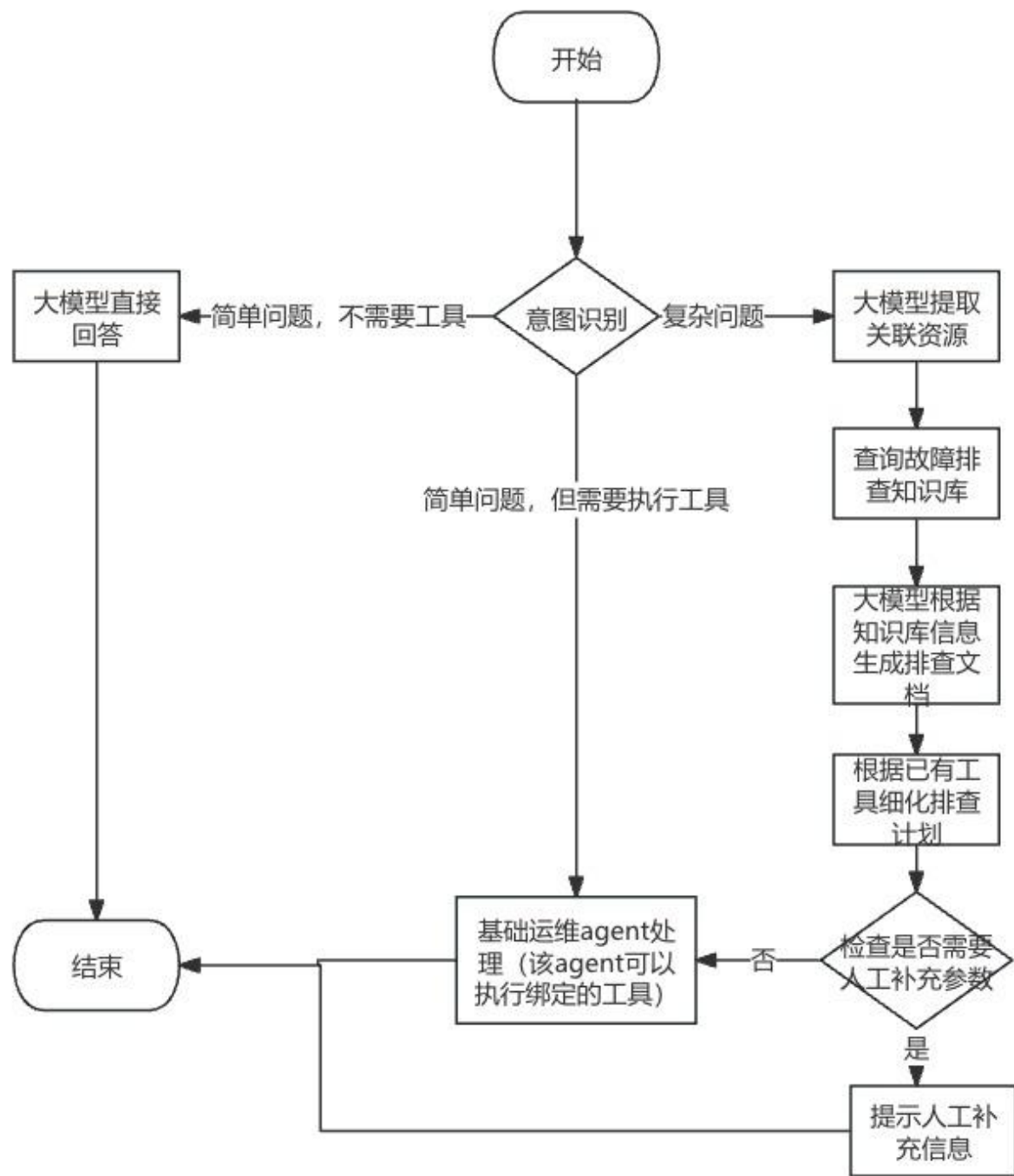
从结果可以看出，系统中存在多次因内存不足导致的进程被杀死的情况。特别是java进程。

内容由AI生成





AI场景 - 告警助手





AI场景 - 自主规划，解决问题（日志分析+告警助手）



- 1. 发现某个主机的某个组件异常
- 2. 生成排查计划

2024-12-17 14:46:53 我

主机172.30.0.134上next_console_service有问题

AI助手 2024-12-17 14:46:57



任务目标

解决172.30.0.134上next_console_service的异常

执行计划

1. 调用日志分析，传入ip和出错的组件

2. 调用告警排查助手，输入上一步中找到的问题

1. 需要按照执行计划执行

👍

🗨

🗑

🔍



AI场景 - 自主规划，解决问题（日志分析+告警助手）



日志分析 2024-12-17 14:47:49



分析172.30.0.134上next_console_service的日志 根据提供的日志信息，可以总结如下：

1. 访问日志 (Access Log)

- 多次出现500错误，请求路径为/api/v1/next_console/frontend_stat，这表明前端统计接口存在异常。
- 也有200状态码的请求，表明部分请求是成功的。

2. 错误日志 (Error Log)

- 多次出现rpc error: code = Unavailable desc = connection error: desc = "transport: Error while dialing: dial tcp 127.0.0.1:8284: connect: connection refused"错误，这表明尝试连接到127.0.0.1:8284的请求失败。
- 这些错误主要出现在tracequeue/tracequeue.go、sqliteexporter/sqlite_exporter.go、logstash/trace_receiver.go和stat/stat_service.go等组件中。

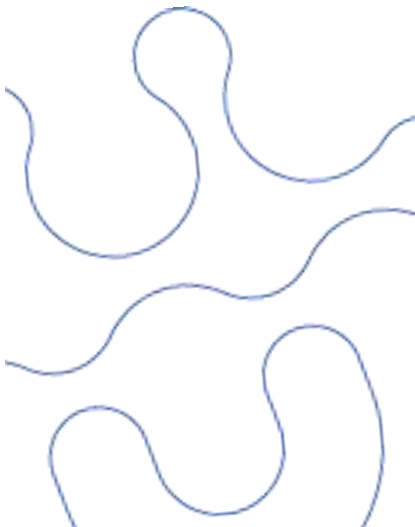
3. OpenTelemetry日志

- 多次出现Exporting failed错误，原因同样是failed to push trace data to console_db: rpc error: code = Unavailable desc = connection error: desc = "transport: Error while dialing: dial tcp 127.0.0.1:8284: connect: connection refused"。
- 这些错误导致数据被丢弃，建议启用retry_on_failure和sending_queue配置选项以提高数据传输的可靠性。

4. 慢查询日志 (Slow Log)

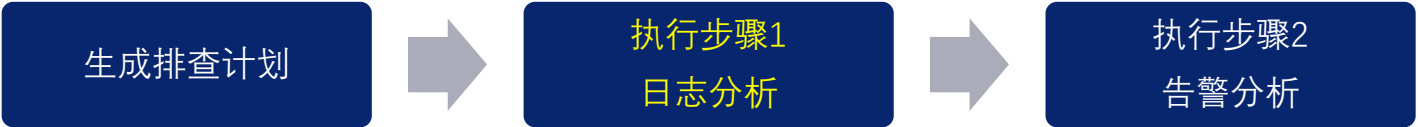
- 没有慢查询日志记录。

3. 进行日志分析





AI场景 - 自主规划，解决问题（日志分析+告警助手）



3. 日志分析得到结论：
8284端口异常

5. 其他错误日志 (Err Log)

- 一些关于协议类型重复注册和组件初始化的警告，这些警告可能不影响系统的主要功能，但建议检查相关配置以避免潜在问题。

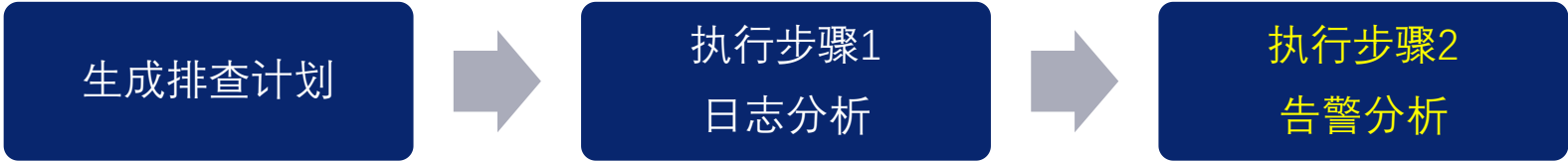
结论

- 主要问题：**127.0.0.1:8284端口无法访问，导致多个组件（如tracequeue、sqliteexporter、logstash和stat）请求失败。
- 建议：**
 - 检查127.0.0.1:8284端口对应的服务是否正常运行。
 - 确认网络配置和防火墙设置，确保端口8284是开放的。
 - 启用retry_on_failure和sending_queue配置选项，以提高数据传输的可靠性。

希望这些信息能帮助您解决问题。如果有更多详细信息或进一步的问题，请随时告知。



AI场景 - 自主规划，解决问题（日志分析+告警助手）



4. 唤起告警助手进行排查：
建议检查console_db日志

AI助手 2024-12-17 14:47:52

日志分析完成，发现问题主机172.30.0.134上8284端口无法访问

继续执行

👍 🗨 📄 🔍

告警排查助手 2024-12-17 14:48:52

🟢 工具调用完成 ▾

主机127.0.0.1上1234端口异常

- ★ 识别意图
- ★ 提取关联资源
- ★ 生成排查计划
- ★ 设置记忆

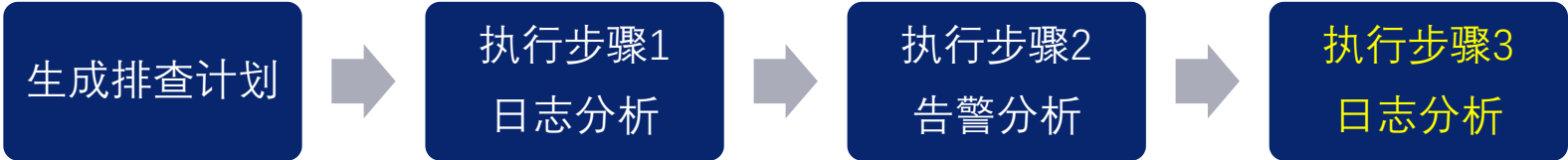
总结结果：

- 进程 console_db 不存在。
- 8284 端口未被监听。
- 未发现系统 OOM 问题。

具体问题：console_db 进程未运行，导致 1234 端口异常。建议检查 console_db 服务的启动配置和日志，以确定具体原因。



AI场景 - 自主规划，解决问题（日志分析+告警助手）



AI助手 2024-12-17 14:48:55



工具调用完成

告警排查完成，已排查127.0.0.1:8284端口问题，为console_db进程未运行导致
继续执行



5. 对console_db进行日志分析

日志分析 2024-12-17 14:49:27



工具调用完成

分析172.30.0.134上console_db的日志 根据172.30.0.134上console_db的日志分析，以下是总结：

Access Log

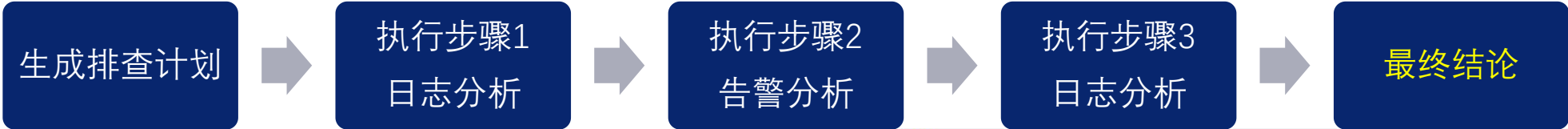
- 最近10条访问日志显示，所有请求均为POST方法，请求路径为/easy_tsdbs.ts_data.ts_data/insert，响应状态码均为200，响应时间在1.1ms到7.6ms之间，没有明显的异常。

Normal Log

- 最近10条ERROR/WARNING日志中，有一条错误信息：
 - 2024-12-17T11:49:06.425+08:00 ERROR easy_sync_cluster::partition::replication:409 | [1] change replica new role:Leader { epoch: 1 }, old role:Leader { epoch: 1 }, other_ids:{}
 - 这条错误信息表明在集群的副本角色变更过程中出现了问题，但具体影响不明确。



AI场景 - 自主规划，解决问题（日志分析+告警助手）



7. 得到最终结论:
console_db的配置config.yaml存在问题

Error Log

- 最近10条错误日志中，有几条启动信息和一条配置文件解析错误：
 - 2024-12-17T11:48:59.911021191+08:00: start easy_tsdb server used=94.11015ms
 - 2024-12-17T11:49:00.280076644+08:00: grpc listen on 0.0.0.0:8285
 - 2024-12-17T11:49:00.422688447+08:00: start init db cluster used=605.936737ms
 - 2024-12-17T11:49:06.461740767+08:00: grpc listen on 0.0.0.0:8284 used=6.644997943s
 - exporter listening on http://0.0.0.0:8286/metrics
 - load config file error,while parsing a block mapping, did not find expected key at line 5 column 1 in bin/../conf/config.yaml
 - 配置文件解析错误可能会影响服务的正常启动和运行。

结论

- 配置文件解析错误: load config file error,while parsing a block mapping, did not find expected key at line 5 column 1 in bin/../conf/config.yaml. 建议检查配置文件config.yaml，确保其格式正确。
- 集群副本角色变更错误: change replica new role:Leader { epoch: 1 }, old role:Leader { epoch: 1 }, other_ids:{}. 建议进一步检查集群状态，确保副本角色变更过程正常。

以上问题可能影响console_db的正常运行，建议尽快处理。

👍

🗨

🔍

🔗

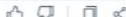
AI助手 2024-12-17 14:49:31



🟢 工具调用完成

已完成，配置文件config.yaml错误

任务完成



AI场景 - 连接外部工具



自动建任务卡片

agent链接报错，创建一个卡片给linuschen

02月13日 15:21

标题: 解决agent链接报错问题

- 故事分类: 体验优化需求
- 产品线: EasyOpsAir
- 功能模块: EasyOps Air
- 估时: 0.2
- 产品经理: linuschen
- 负责人: linuschen
- 报告人: linuschen
- 故事描述: 解决agent链接报错问题, 确保链接正常工作
- 验收标准: agent链接不再报错, 用户可以正常访问

确认

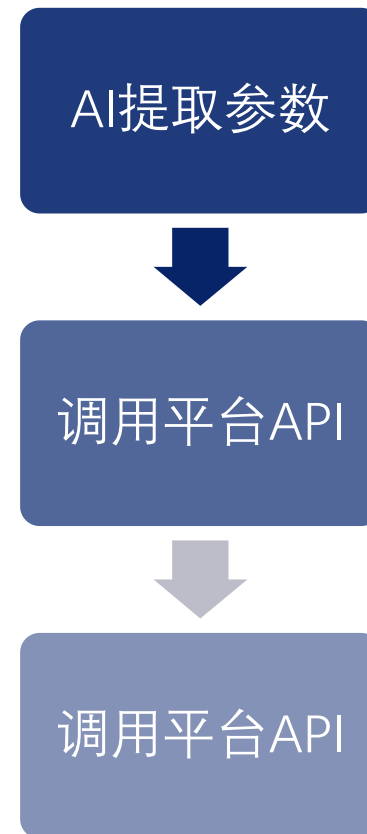
提示: 你需要点击"确认"按钮才会真的创建故事, 如果要修改不正确的地方, 请说修改xxxx

内容由AI生成

负责人为winsen

标题: 解决agent链接报错问题

- 故事分类: 体验优化需求
- 产品线: EasyOpsAir
- 功能模块: EasyOps Air
- 估时: 0.2
- 产品经理: linuschen





3
—

AI开发



AI开发



让每个人都拥有个人助理

超级会话终端

打通运维工具体系，
结合上下文，AI无
处不在。
打通IM，日常工作
日常交流

大模型服务

分离上层应用和算
力建设，可对接本
地算力资源也可对
接SaaS服务

微调训练

针对特定场景，提
供训练集管理和大
模型微调能力，让
大模型在垂直领域
更聪明



知识库管理

提供私有知识库向
量化检索

智能体建设

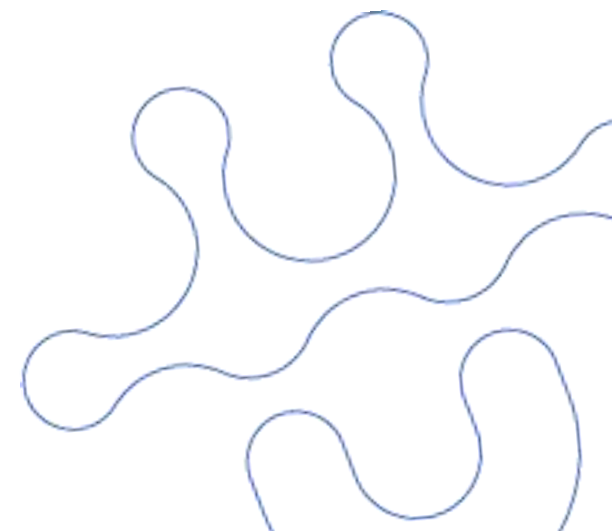
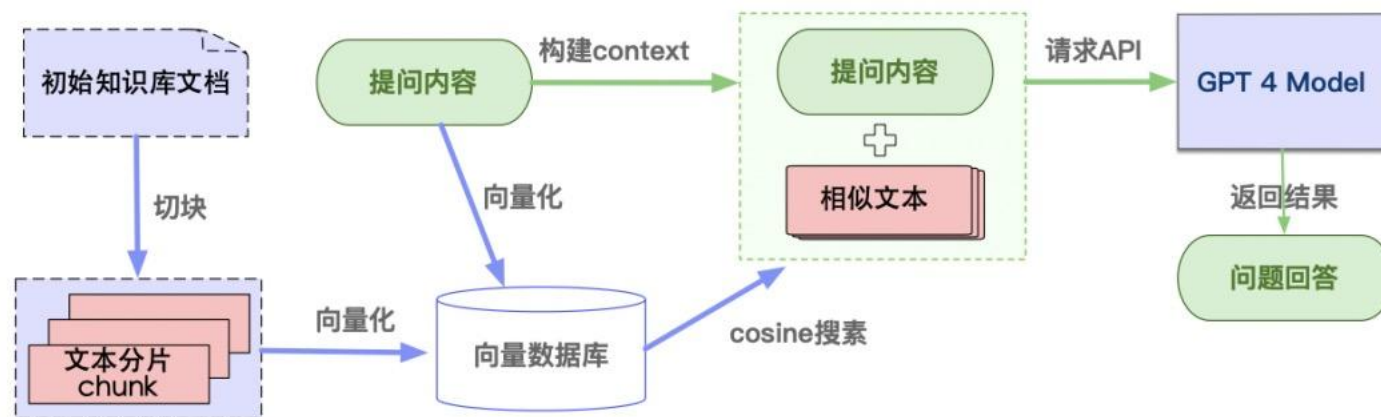
根据业务场景，创
建专属智能体



- 知识库向量化



- 查询





本地知识库检索 - 支持多种知识库



支持多种知识库的导入，让知识发挥他的最大价值



对接更多知识



在现代工作环境中，知识的积累是企业的重要资产，它存在于各种形式，如专业知识库、文档、工单记录和日常对话中。然而，当这些知识仅存于个别人员手中，或检索效率低下时，它们的价值无法充分释放。

现在有了大模型的技术助力，这些问题都可以得到很好的解决：

- * 知识整合：通过先进的数据管理和分析工具，我们可以将分散的知识系统化，使其更容易访问和理解。
- * 智能检索：利用自然语言处理和机器学习，大模型可以提供精准的搜索功能，让所需信息瞬间呈现。
- * 知识共享：借助协作平台，知识可以跨越个人和团队的界限，实现全组织的共享和交流。
- * 自动化学习：大模型还能根据用户的使用习惯和需求，持续学习和优化，提供个性化的学习和参考资源。

如下说明各种对接方法：

语雀 Confluence Jira 流程工单 IM会话 其他

获取知识方法：

- 可通过语雀的接口获得：
- 获取指定知识库列表的全部文档，再循环获得每个文档的内容。
 - 之后再调统一的文档插入接口插入到当前知识库

点击 [开始配置](#)

从CMDB来 - 单表

名称

easy_cmdb系统的主机

选择模型

主机(HOST)

过滤条件

全部实例

指定实例

动态过滤器

查询语句

查询语句

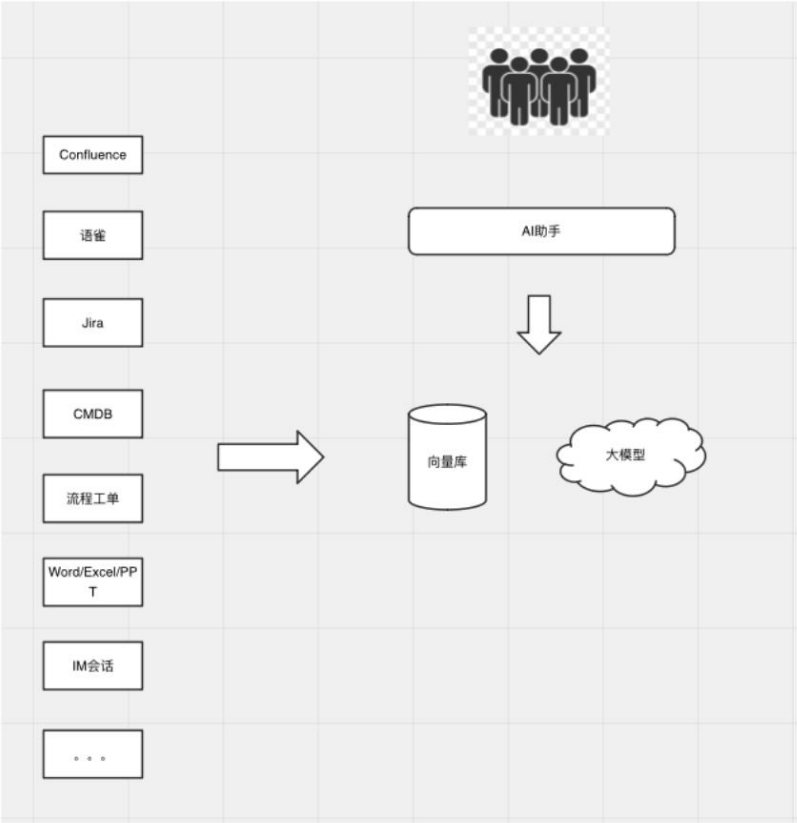
返回字段

hostname x ip x owner.name x 添加

数据预览 (最多10行):

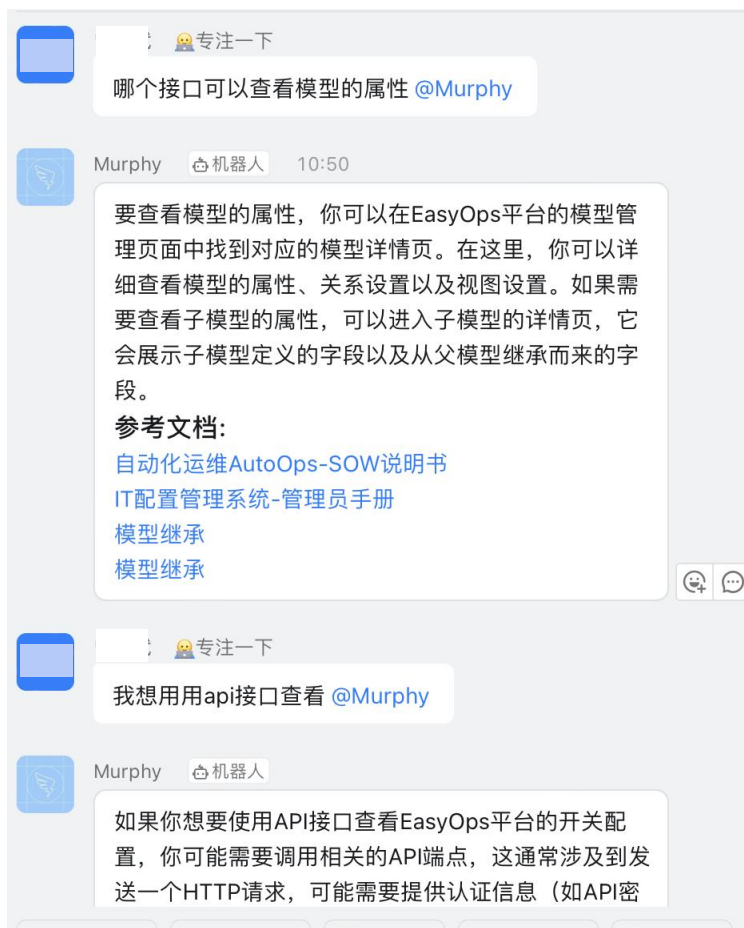
切换到JSON

_object_id	hostname	instanceId	ip	owner
HOST	ops-host-5-10-nginx	59ab09141c690	10.0.5.10	[{"instanceId":"56fdd9c880r"
HOST	ops-physical-100-110	59ab0f092049a	192.168.100.110	[{"instanceId":"5cb51f92df034
HOST	ops-physical-100-115	59abf0fa135ec	192.168.100.115	[{"instanceId":"5cb51f92df034





知识闭环，知识需要有反馈才能让不断
提升其准确性，**越用越好用**



提问者标记

大模型分析

会话上下文感知



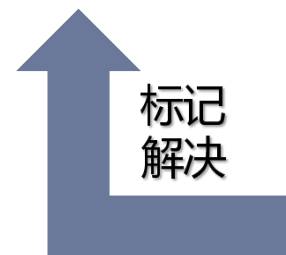
疑似待完善词条



人为优化



更新知识库



标记解决



Agent编排





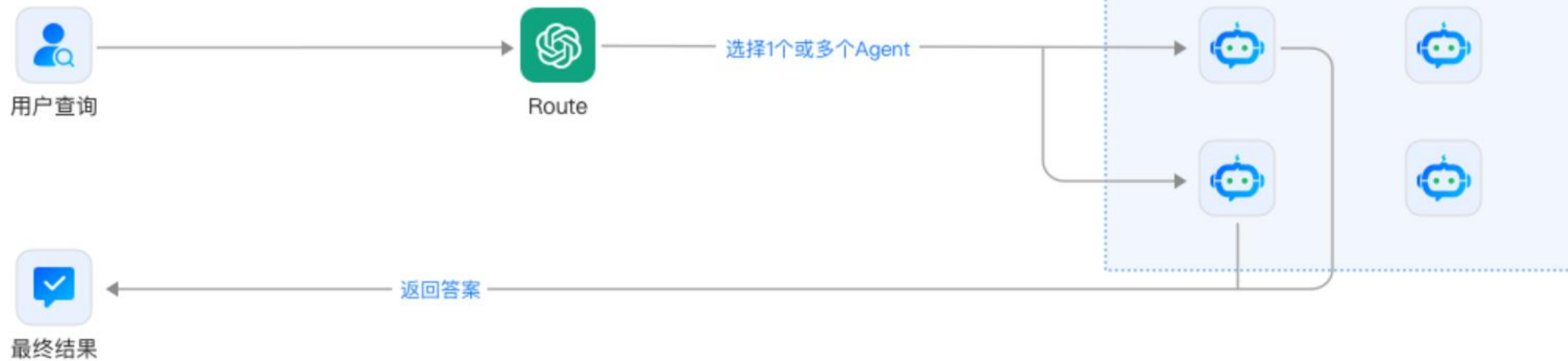
AI助手



单聊模式



群聊模式

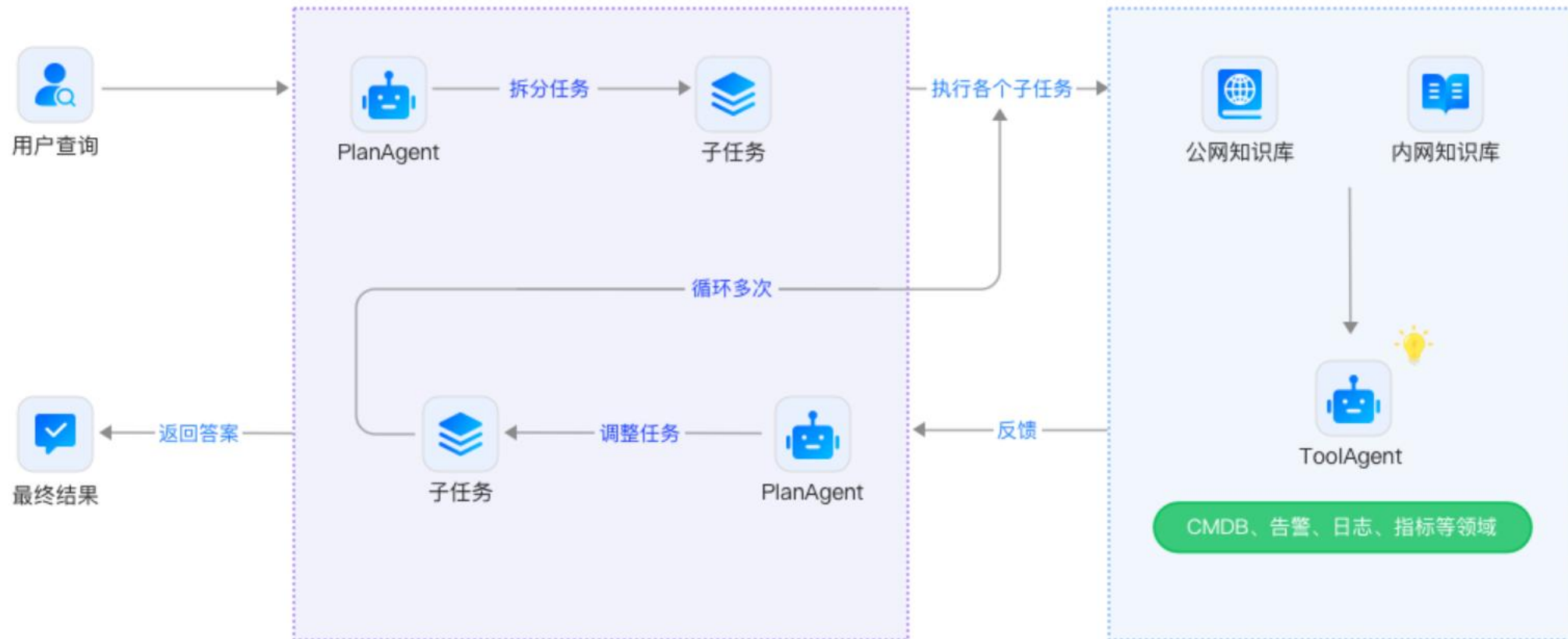




AI助手



任务计划





AI开发 - 技术架构





THANK YOU

连接IT和业务，让IT为业务提供持续创新动能



优维科技（深圳）有限公司