

如何构建 MultiAgent

Eino ADK与A2A实践

演讲人：王德政

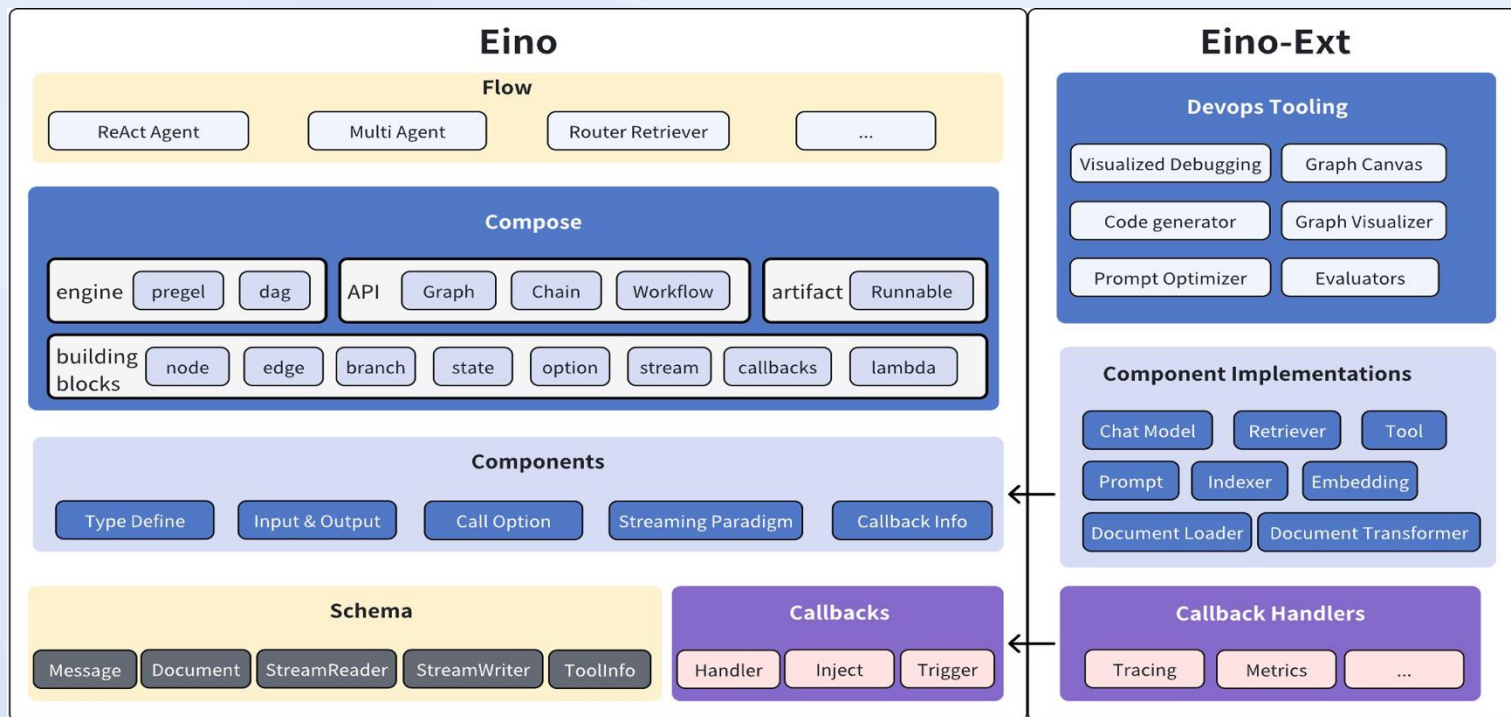
SPEAKER: WangDeZheng



Eino

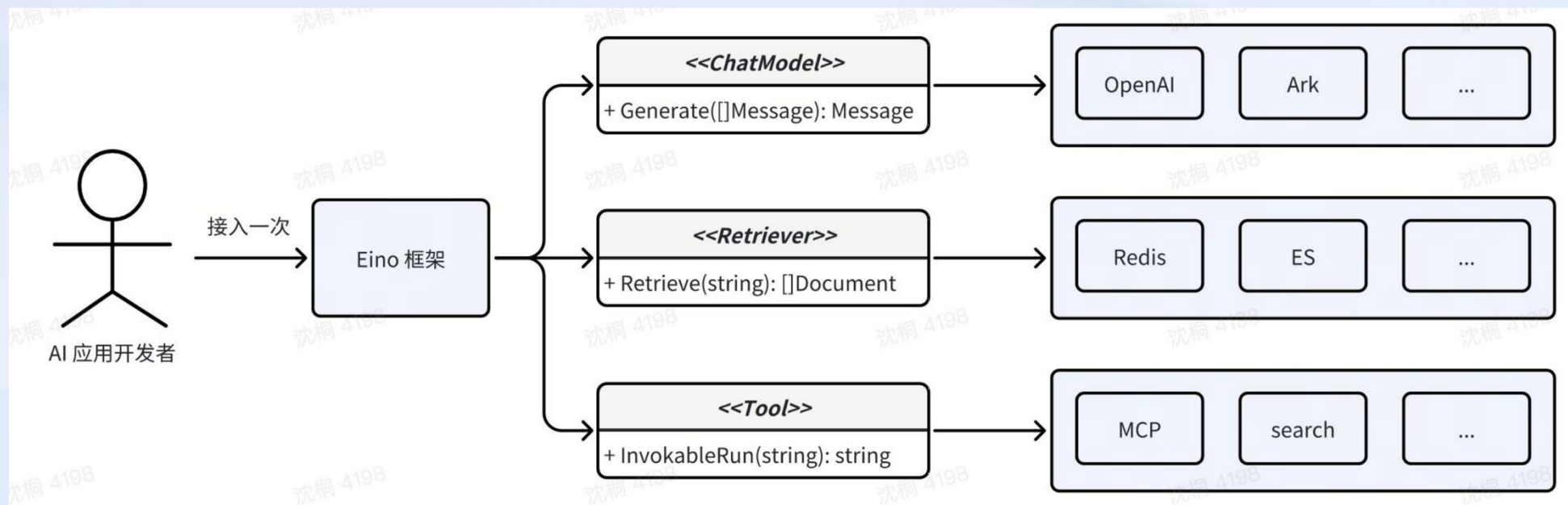
Eino['aino] 是一个 Golang 语言的开源 AI 应用开发框架。 它从开源社区中的诸多优秀 LLM 应用开发框架，如 LangChain、LangGraph 和 LlamaIndex 等获取灵感，提供了一个强调简洁性、可扩展性、可靠性与有效性，且更符合 Go 语言编程惯例的 AI 应用开发框架。

目前 Eino 在 Github 上已经收获 7.3k Star，并在公司内部500+服务、目前已知十余家外部企业落地。



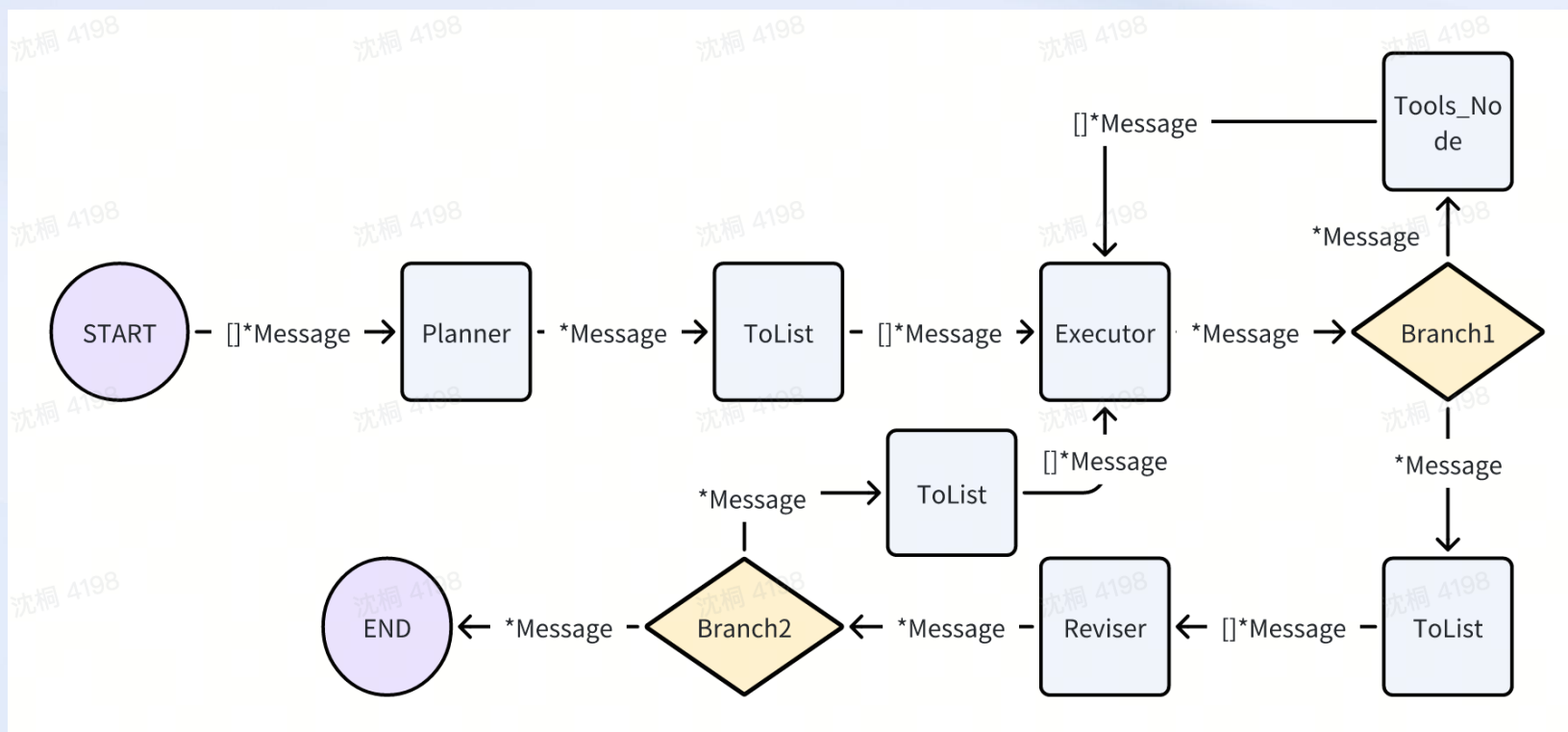
Eino

提供了一系列 组件（component） 抽象与实现，可轻松复用与组合，用于构建 LLM 应用。



Eino

提供了 Graph 编排能力，将各种各样的组件串联起来。



目录 | Contents

Part 01 Eino ADK 介绍

- 为什么需要ADK
- 什么是ADK
- Agent 抽象
- 多 Agent 协作
- 开箱即用的 Agents

Part 02 A2A 介绍

- 什么是A2A
- 在Eino中使用A2A

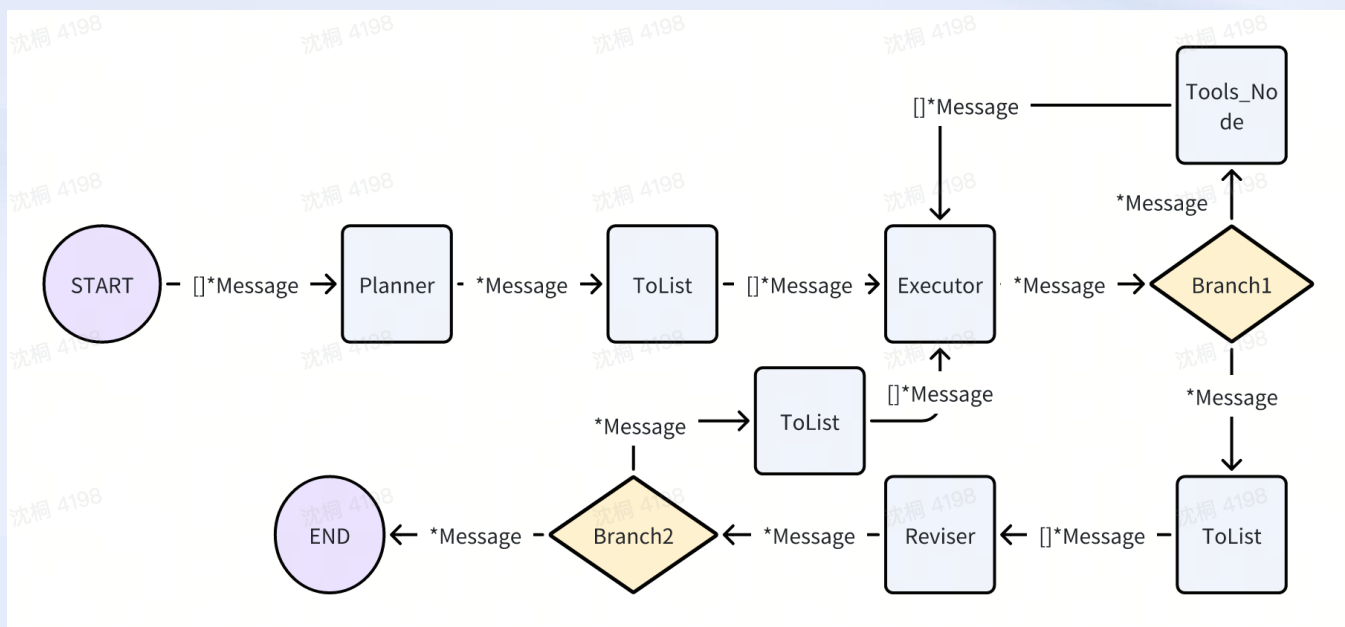
Part 03 示例

01

Eino ADK介绍

为什么需要 ADK

Eino Graph 提供了基于组件的编排能力：

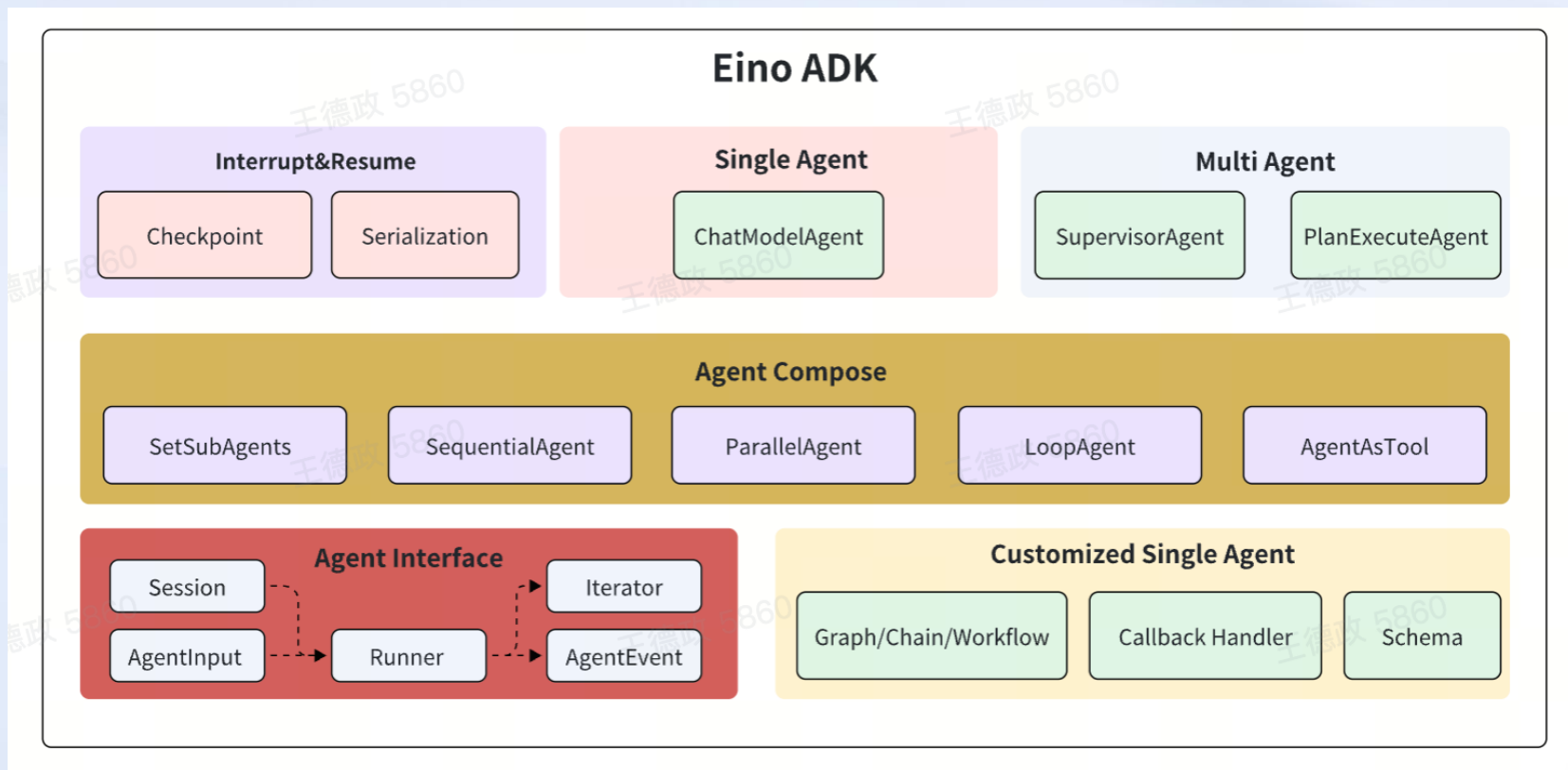


过高灵活性使易用性下降：

- 类型转换、节点依赖关系、状态管理复杂
- 缺乏对Agent原生结构（如记忆）的直接支持

什么是 ADK

Eino ADK (Agent Development Kit) 参考了Google ADK, 提供了Agent 级别的抽象、多 Agent 构建能力以及上下文传递、中断与恢复等功能, 同时复用了Eino的组件生态, 让开发者更简单、直观地构建 Go Agent 应用。



Agent 抽象

Eino ADK 的核心是 Agent 抽象(Agent Interface), ADK 的所有功能设计均围绕 Agent 抽象展开。

```
type Agent interface {  
    Name(ctx context.Context) string  
    Description(ctx context.Context) string  
    Run(ctx context.Context, input *AgentInput, options ...AgentRunOption) *AsyncIterator[*AgentEvent]  
}
```

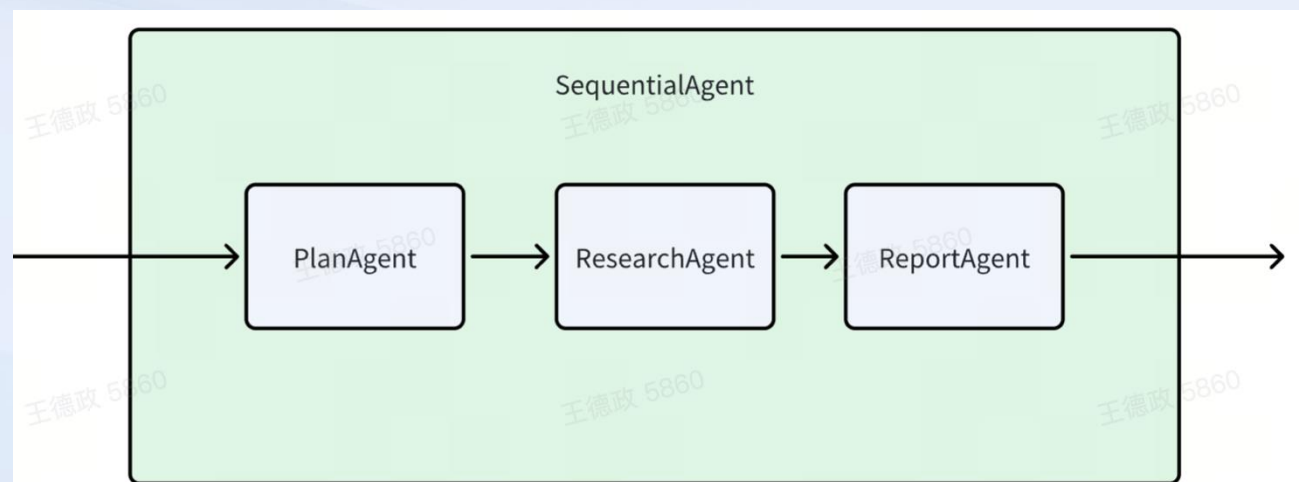
- Name 与 Description 定义了 Agent 的身份与能力, 为多 Agent 之间的自动化协作提供了支持
- Run 方法定义了与 Agent 交互的统一方式, AgentInput 包含 Agent 的输入, AsyncIterator 能够实时返回 Agent 中产生 AgentEvent, AgentEvent 中包含了 Agent 输出、指令、报错等信息

多 Agent 协作

Workflow

ADK 提供了以固定顺序运行多个 Agent 的能力：

- Sequential: 以固定顺序执行多个 Agent



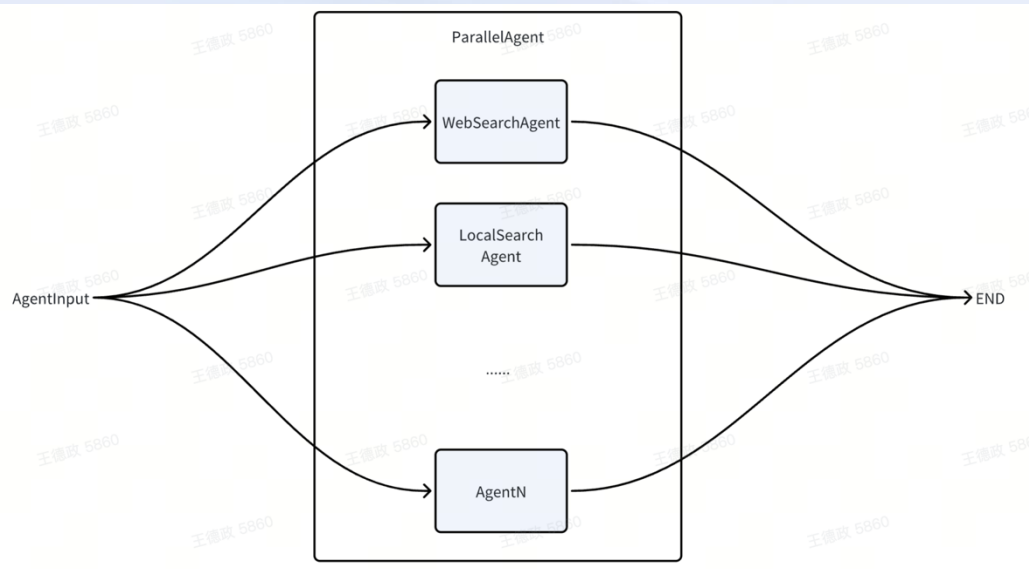
```
seqAgent, err := adk.NewSequentialAgent(ctx, &adk.SequentialAgentConfig{
    Name:      "xxx",
    Description: "xxx",
    SubAgents: []adk.Agent{agent1, agent2, agent3},
})
```

多 Agent 协作

Workflow

ADK 提供了以固定顺序运行多个 Agent 的能力：

- Parallel：并行执行多个 Agent



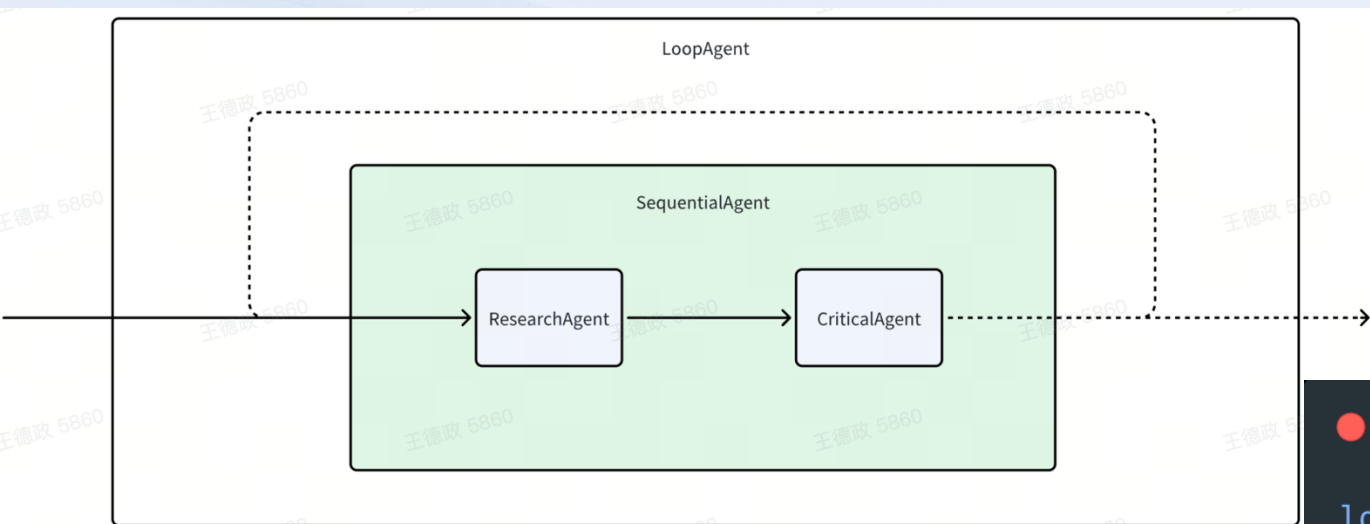
```
parallelAgent, err := adk.NewParallelAgent(ctx, &adk.ParallelAgentConfig{
    Name:      "xxx",
    Description: "xxx",
    SubAgents: []adk.Agent{agent1, agent2, agent3},
})
```

多 Agent 协作

Workflow

ADK 提供了以固定顺序运行多个 Agent 的能力：

- Loop：循环执行多个 Agent

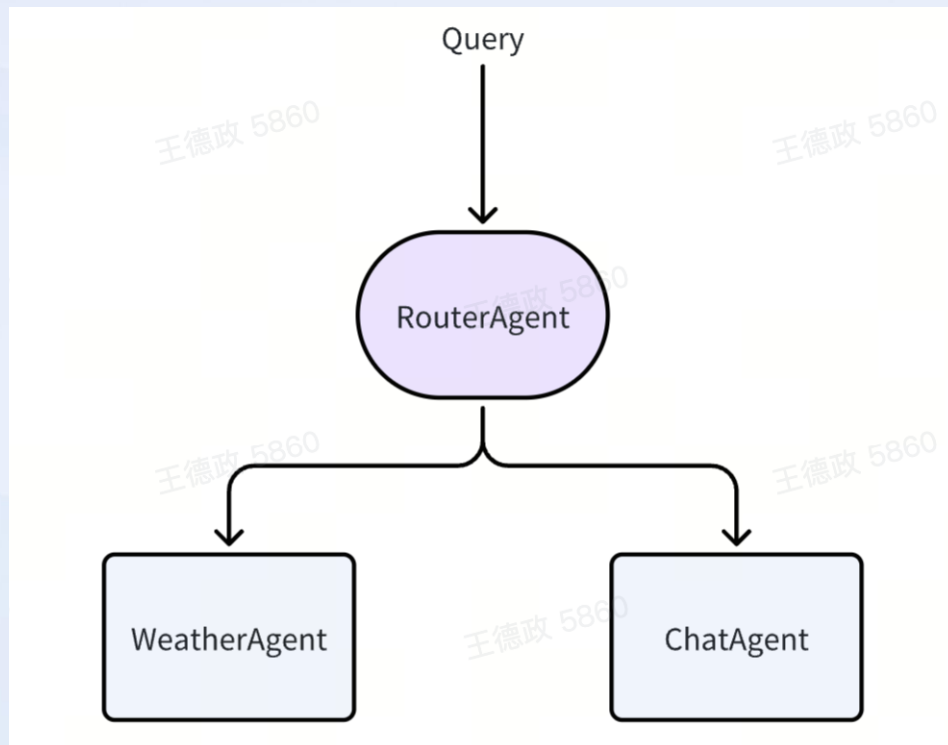


```
loopAgent, err := adk.NewLoopAgent(ctx, &adk.LoopAgentConfig{
    Name:          "xxx",
    Description:    "xxx",
    SubAgents:     []adk.Agent{agent1, agent2, agent3},
})
```

多 Agent 协作

Transfer

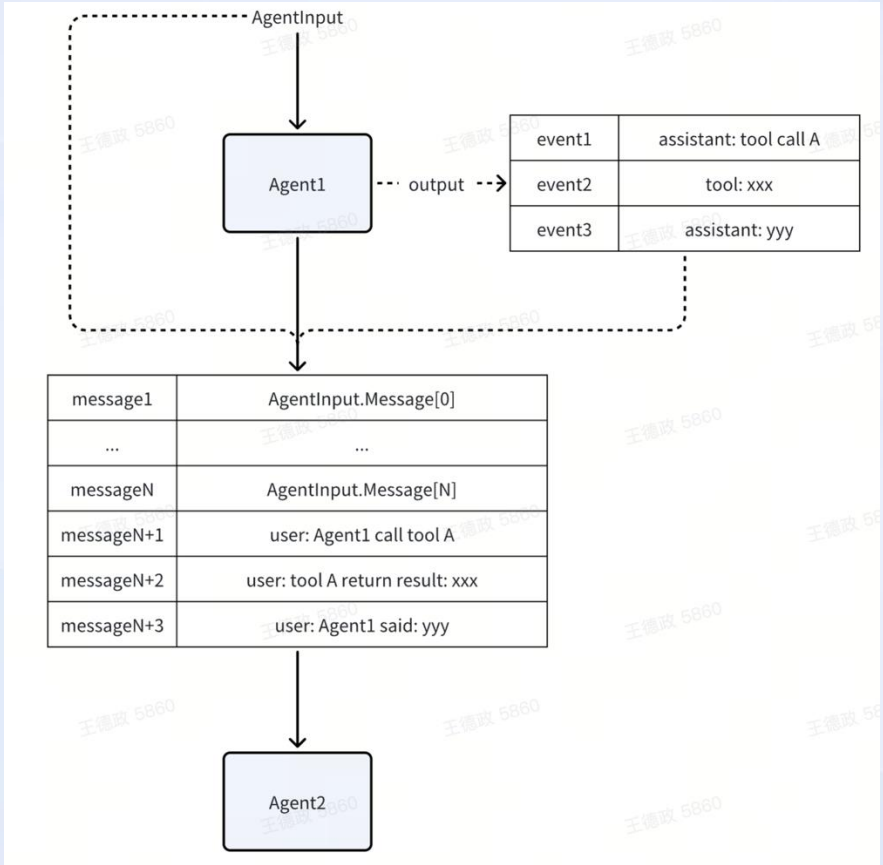
Agent 产生包含 Transfer 指令的 Event 后，ADK 框架会调用目标 Agent，利用这个机制可以在运行时动态决定 Agent 路由：



多 Agent 协作

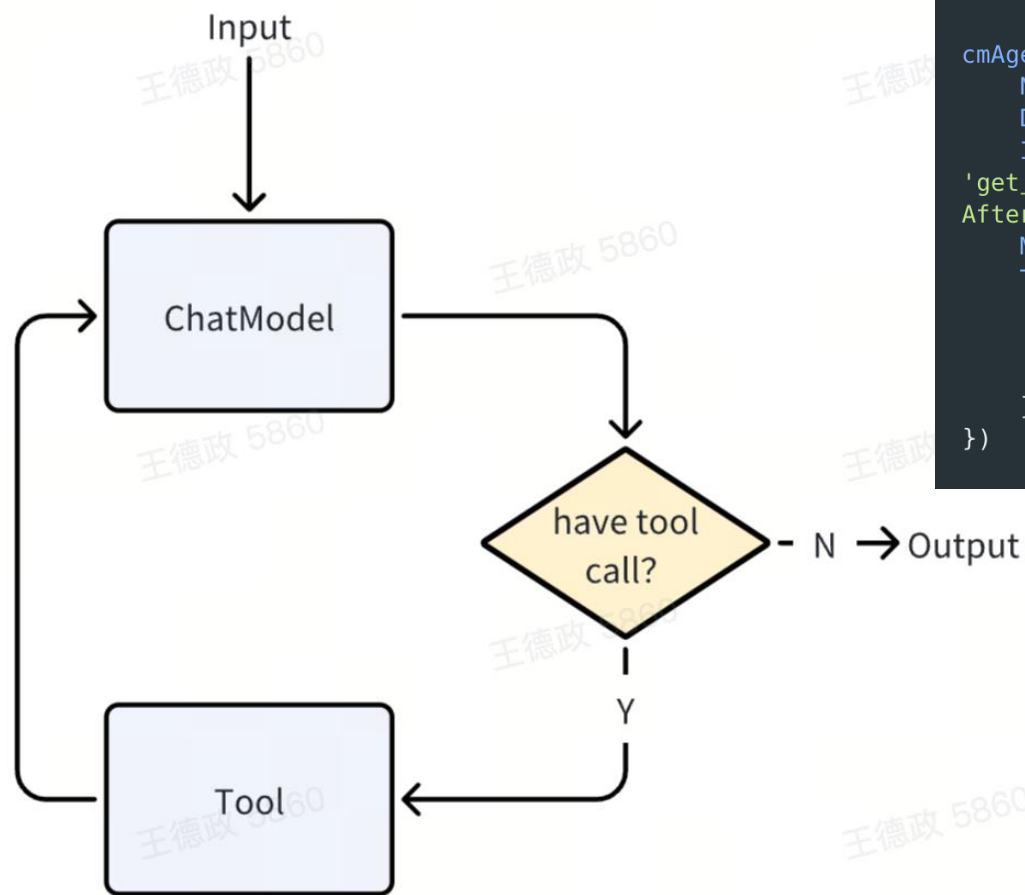
上下文传递

ADK 中每个 Agent 产生的 AgentEvent 都会保存在 Session 中，一个新的 Agent 运行时，这些 Event 会转化成这个 Agent 的输入，ADK 提供了默认转换方式：



开箱即用的 Agents

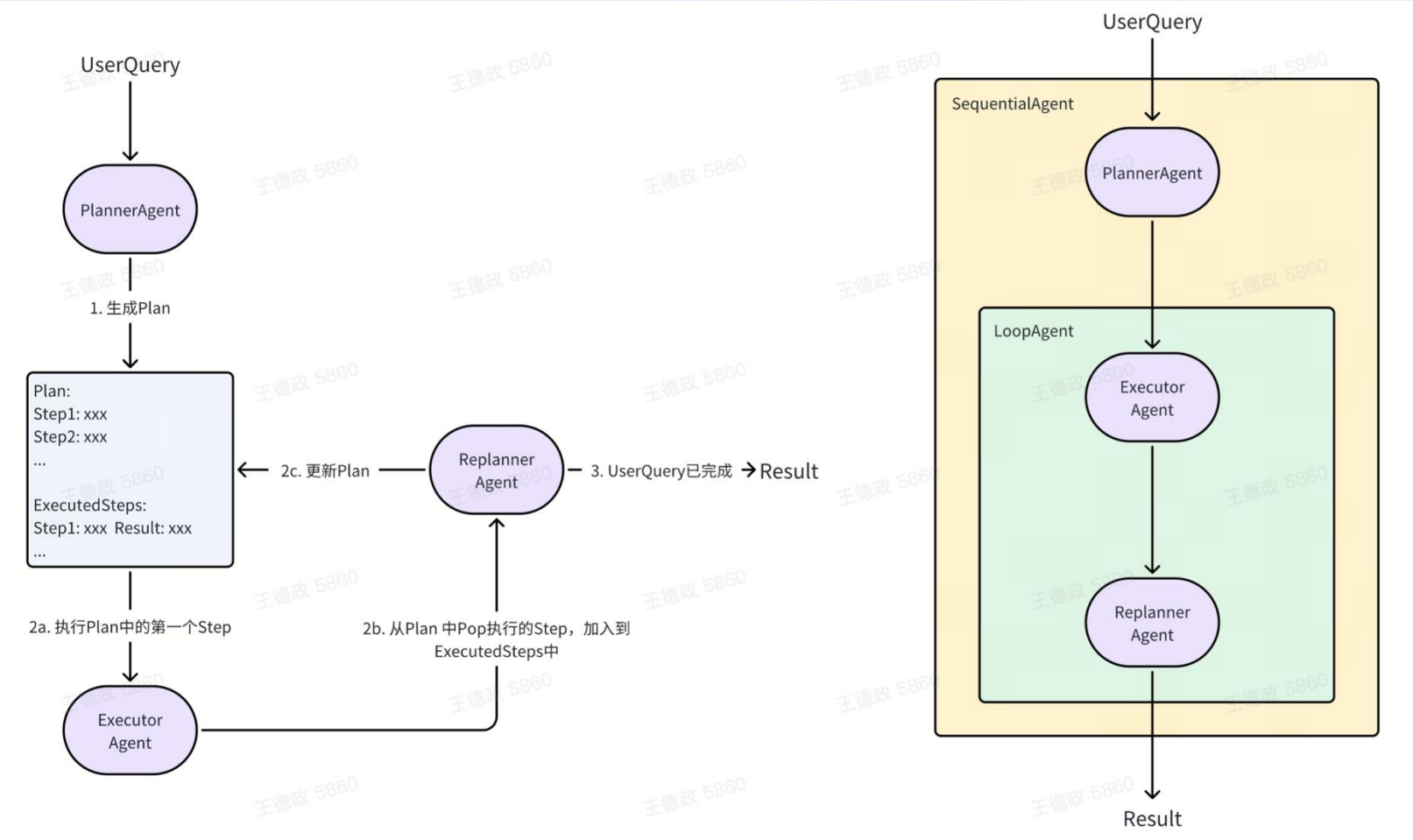
ChatModelAgent (ReactAgent)



```
cmAgent, err := adk.NewChatModelAgent(context.Background(), &adk.ChatModelAgentConfig{
    Name:      "WeatherAgent",
    Description: "This agent can get the current weather for a given city.",
    Instruction: `Your sole purpose is to get the current weather for a given city by using the
'get_weather' tool.
After calling the tool, report the result directly to the user.` ,
    Model: model.NewChatModel(),
    ToolsConfig: adk.ToolsConfig{
        ToolsNodeConfig: compose.ToolsNodeConfig{
            Tools: []tool.BaseTool{weatherTool},
        },
    },
})
```


开箱即用的 Agents

PlanAndExecute



开箱即用的 Agents

PlanAndExecute



```
pneAgent, err := planexecute.New(ctx, &planexecute.Config{
    Planner:      planexecute.NewPlanner(ctx, &planexecute.PlannerConfig{xxx}),
    Executor:     planexecute.NewExecutor(ctx, &planexecute.ExecutorConfig{xxx}),
    Replanner:    planexecute.NewReplanner(ctx, &planexecute.ReplannerConfig{xxx}),
    MaxIterations: 20,
})
```

开箱即用的 Agents

PlanAndExecute



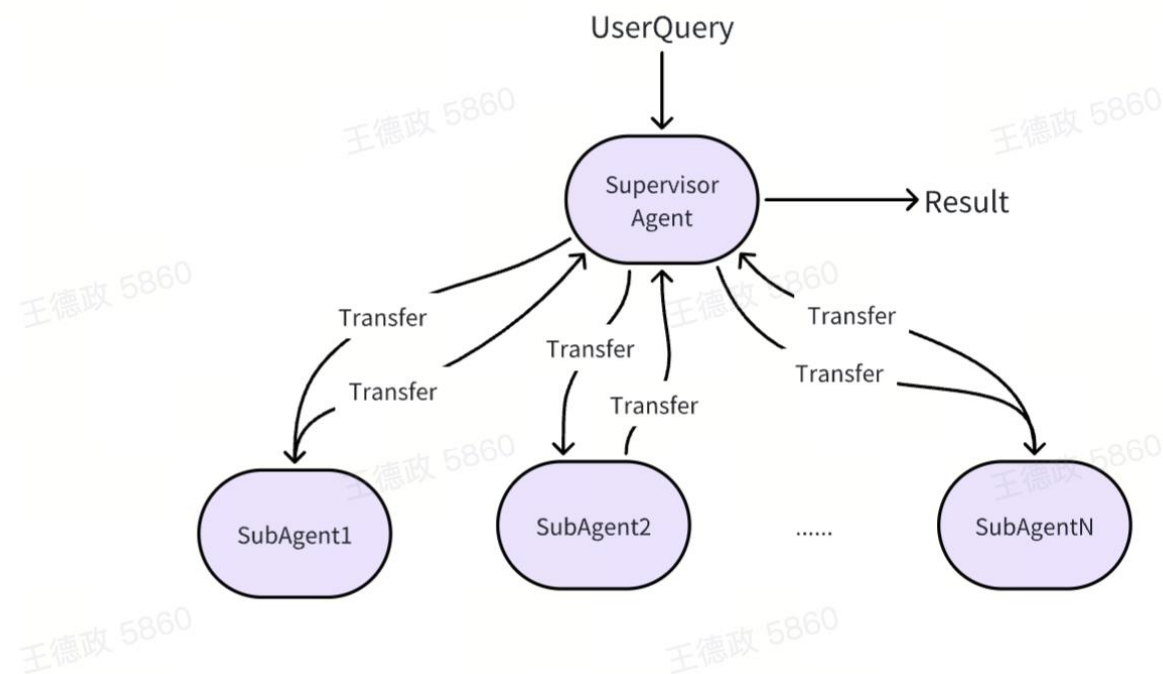
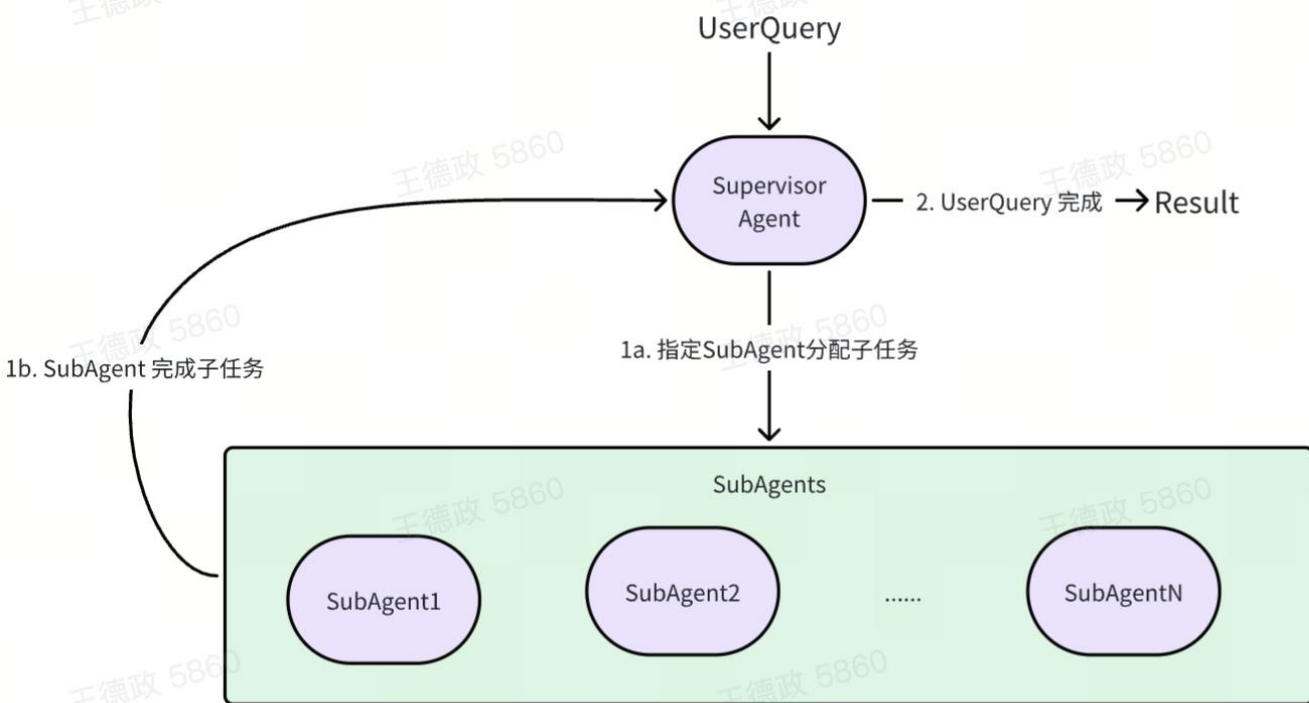
```
type Config struct {
    Planner adk.Agent
    Executor adk.Agent
    Replanner adk.Agent
    MaxIterations int
}

type PlannerConfig struct {
    ToolCallingChatModel model.ToolCallingChatModel
    ToolInfo *schema.ToolInfo
    GenInputFn GenPlannerModelInputFn
    Factory PlanFactory
}
```

PlannerAgent	ChatModelAgent	ChatModel
		PlanTool
		PlanDef
		Prompt
ExecutorAgent	ChatModelAgent	ChatModel
		Tools
ReplannerAgent	ChatModelAgent	Prompt
		ChatModel
		Tools
		Prompt

开箱即用的 Agents

Supervisor



开箱即用的 Agents

Supervisor

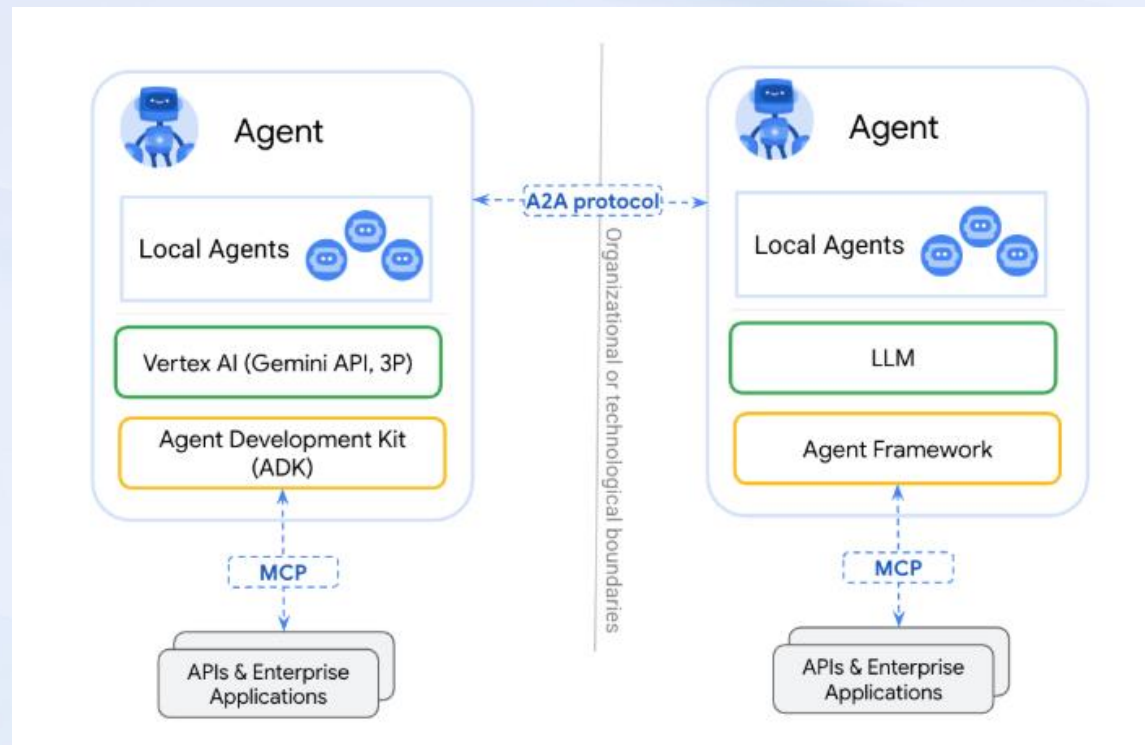


```
supAgent, err := supervisor.New(ctx, &supervisor.Config{  
    Supervisor: adk.NewChatModelAgent(ctx, &adk.ChatModelAgentConfig{xxx}),  
    SubAgents:  []adk.Agent{agent1, agent2, agent3},  
})
```

02 | A2A 介绍

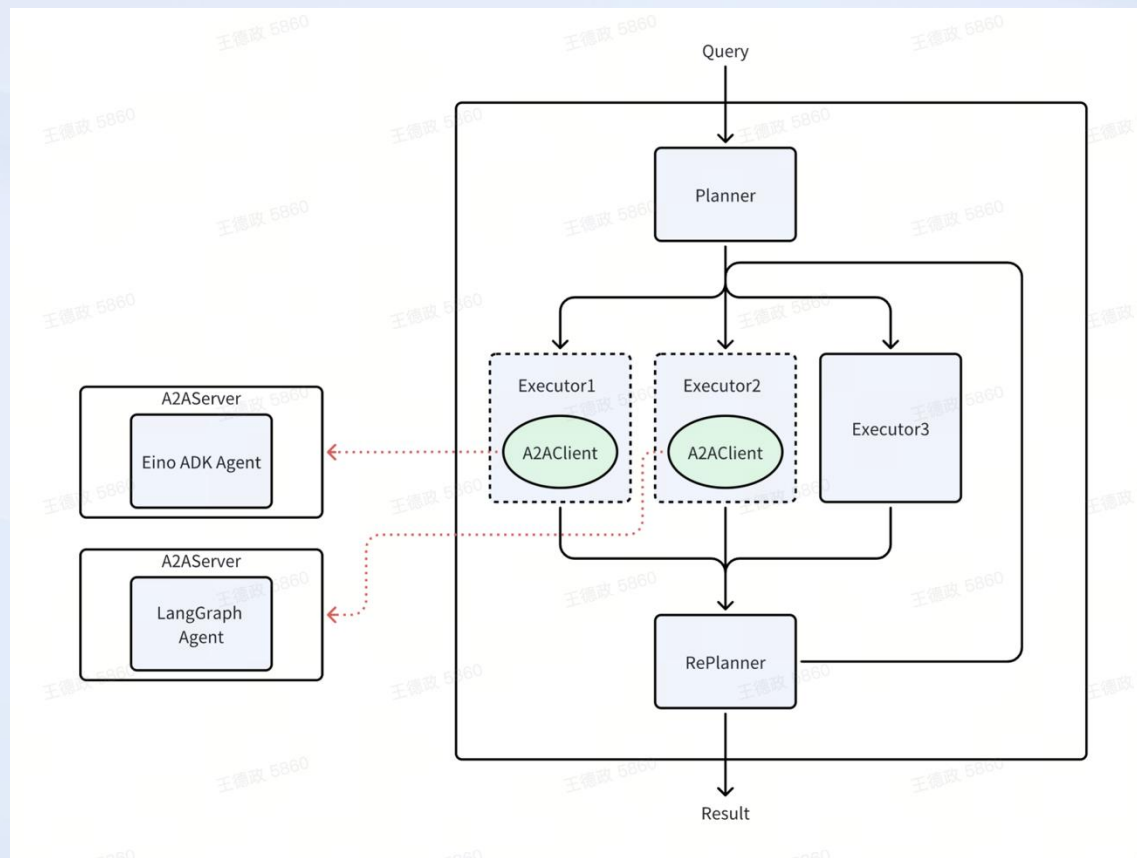
什么是 A2A

A2A (AgentToAgent) 协议是 Google 提出一种开放标准, 旨在实现 AI Agent 之间的无缝通信与协作。



A2A 在 Eino 中的应用

Eino ADK 提供了双向封装能力：既能将 Agent 发布为 A2A 服务，也能将 A2A 服务转换为本地 Agent，让开发者无需关心底层协议。



03 | 示例

TravelResearchAgent

场景说明

开发一个旅游规划 Agent，结合天气、机酒、目的地吸引力等角度生成一份旅游计划。


Agent 结构设计

背景信息调研产生的上下文较大，且不同项目调研之间上下文不应共享，所以才用PlanAndExecute的方式实现此Agent：

1. Planner：生成制定旅游计划的步骤
2. Executor：使用多种工具（天气查询、航班酒店搜索、目的地吸引力）执行计划
3. Replanner：评估执行结果，若信息不足则调整计划，否则生成最终总结

TravelResearchAgent

创建PlanAndExecute Agent之前，需要先创建合适的Planner、Executor和Replanner：




```
entryAgent, err := planexecute.New(ctx, &planexecute.Config{
    Planner:      planAgent,
    Executor:     executeAgent,
    Replanner:    replanAgent,
    MaxIterations: 20,
})
```

[代码地址](#)

TravelResearchAgent

这里直接使用默认实现创建 Planner 和 RePlanAgent, 只需配置ChatModel:



```
planAgent, err := planexecute.NewPlanner(ctx, &planexecute.PlannerConfig{
    ToolCallingChatModel: model.NewChatModel(),
})
replanAgent, err := planexecute.NewReplanner(ctx, &planexecute.ReplannerConfig{
    ChatModel: model.NewChatModel(),
})
```

TravelResearchAgent

Executor 需要自定义 Prompt 来优化工具调用效果:



```
schema.SystemMessage(`You are a diligent and meticulous travel research executor, Follow the given  
plan and execute your tasks carefully and thoroughly.  
Execute each planning step by using available tools.  
For weather queries, use get_weather tool.  
For flight searches, use search_flights tool.  
For hotel searches, use search_hotels tool.  
For attraction research, use search_attractions tool.  
For user's clarification, use ask_for_clarification tool. In summary, repeat the questions and results  
to confirm with the user, try to avoid disturbing users'  
Provide detailed results for each task.  
Cloud Call multiple tools to get the final result.`),
```

TravelResearchAgent

通过 GenInput 配置自定义 Prompt 并渲染:

```
executeAgent, err := planexecute.NewExecutor(ctx, &planexecute.ExecutorConfig{
    Model:      model.NewChatModel(),
    ToolsConfig: adk.ToolsConfig{ToolsNodeConfig: compose.ToolsNodeConfig{Tools: travelTools}},

    GenInputFn: func(ctx context.Context, in *planexecute.ExecutionContext) ([]adk.Message, error){
        planContent, err_ := in.Plan.MarshalJSON()
        if err_ != nil {return nil, err_}
        firstStep := in.Plan.FirstStep()
        msgs, err_ := executorPrompt.Format(ctx, map[string]any{
            "input":      formatInput(in.UserInput),
            "plan":        string(planContent),
            "executed_steps": formatExecutedSteps(in.ExecutedSteps),
            "step":        firstStep,
        })
        if err_ != nil {return nil, err_}
        return msgs, nil
    },
})
```


TravelResearchAgent

Trace

plan-execute-replan

Success

Latency: 256663 ms Tokens:139,003

调用树

plan-execute-replan

4.28min

DefaultChatTemplate

prompt

0ms

OpenAIChatModel

model

13.04s

795

DefaultChatTemplate

prompt

0ms

React

6.34s

ChatModel

model

1.94s

873

ToolNode

0ms

search_flights

0ms

ChatModel

model

4.40s

1262

DefaultChatTemplate

prompt

0ms

OpenAIChatModel

model

3.26s

1535

DefaultChatTemplate

prompt

0ms

React

5.55s

ChatModel

model

5.55s

1481

DefaultChatTemplate

prompt

0ms

OpenAIChatModel

model

2.73s

1888

DefaultChatTemplate

prompt

0ms

plan-execute-replan

标注数据

Run

Metadata

Feedback

Input

TEXT

JSON

Plan a 3-day trip to Beijing in Next Month. I need flights from New York, hotel recommendations, and must-see attractions.

Today is 2025-09-09. All the details are up to you.

Output

TEXT

JSON

{ 3 Items

"role": "assistant"

"content":

"For your 3-day trip to Beijing, the following updated flights are available from New York (JFK) to Beijing (PEK) and vice versa: ### Outbound Flights (New York to Beijing on October 15, 2025): 1. **China Eastern** - **Flight No**:

Ch1075 - **D

eparture**:

05:33 - **Arrival**:

23:27 - **...

"response_meta": { 2 Items

"finish_reason": "stop"

"usage": { 4 Items

"prompt_tokens": 6621

"prompt_token_details": { 1 Items

"cached_tokens": 6272

"completion_tokens": 573

"total_tokens": 7194

Status

Success

StatusCode

-

SpanID

1a22422c7bc3c...

Type

custom

Latency

256,663ms

StartTime

2025-09-17 21:45:40

TravelResearchAgent

利用 Eino 提供的封装，将创建好的 TravelResearchAgent 以 A2AServer 的形式提供服务：

```
h := hertz_server.Default()
r, err := jsonrpc.NewRegistrar(ctx, &jsonrpc.ServerConfig{
    Router:      h,
    HandlerPath: "/travel_agent",
})
if err != nil {
    log.Fatal(err)
}
err = eino.RegisterServerHandlers(ctx, entryAgent, &eino.ServerConfig{
    Registrar: r,
})
if err != nil {
    log.Fatal(err)
}

_ = h.Run()
```

TravelResearchAgent

也可以将 A2AClient 封装成 Agent 供 TravelResearchAgent 使用:



```
t, err := jsonrpc.NewTransport(ctx, &jsonrpc.ClientConfig{
    BaseURL:      "xxx",
    HandlerPath:  "xxx",
})
executeAgent, err := eino.NewAgent(ctx, eino.AgentConfig{
    Transport: t,
})
if err != nil {
    log.Fatal(err)
}
```

THANKS

