QCon
全球软件开发大会

InfoQ 极客传媒

# AI CodeReview 实践：代码变更阶段的风险识别与阻断

演讲人：严宽

哔哩哔哩/资深测试开发工程师

# 目录

InfoQ 极客传媒

北京

QCon 全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AIOps
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

北京

AiCon 全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

上海

QCon 全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月 — 5月 — 6月 — 8月 — 10月 — 12月

上海

AiCon 全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

深圳

AiCon 全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

北京

AiCon 全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LMOps
- 具身智能

# 极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播

参会咨询

查看会议

# 01

# 代码审查及行业现状

# 传统代码变更流程


开发者提交代码
开发者发起Pull Request/Merge Request
自动化测试
人工代码审查
通过CI流水线的方式执行一系列的自动化任务对变更进行评估测试
同时有其它开发者完成本次变更的人工代码审查
反馈与修改
代码提交人针对上一步流水线的执行情况以及代码审查人员的审查意见进行修改，并重复提交直到认定审查通过
代码合并
代码变更合入到主分支

# 代码审查的价值与痛点

- 风险阻断与拦截
- 统一标准规范
- 知识共享
- 代码质量保障


- 耗时耗力
- 主观性强
- 知识壁垒
- 低效/流于形式

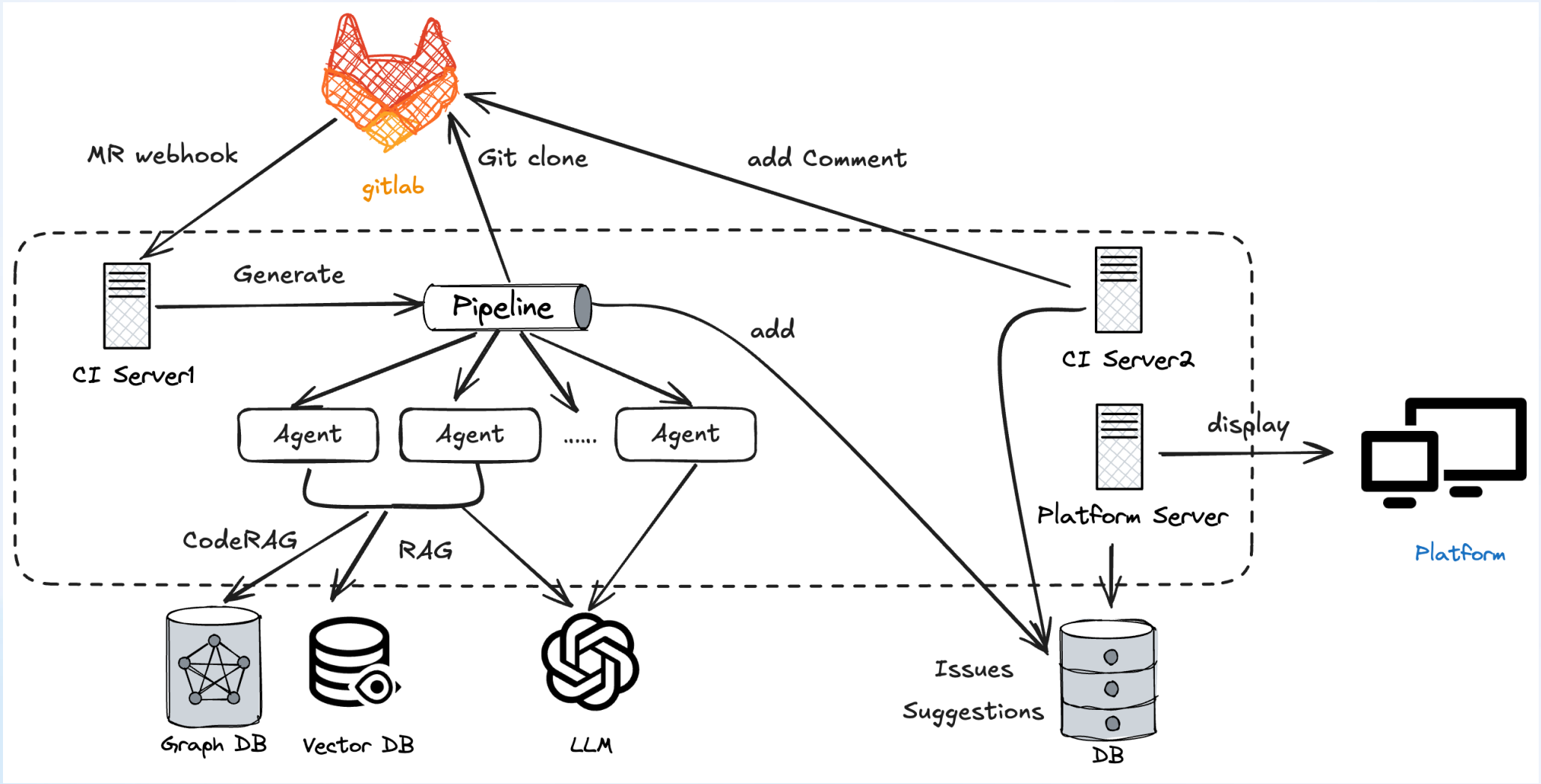
# 大模型时代的代码审查


- 模型
  - 国外：
    - Claude Opus 4
    - Claude Sonnet 4
    - Gemini 2.5 Pro
    - GPT-5
  - 国产：
    - Qwen3-Coder
    - Deepseek-R1
    - Kimi-K2
- 功能
  - PR描述
  - 变更评审
  - 代码建议
  - chat
- IDE/CLI产品
  - 国外：
    - Claude Code
    - Gemini CLI
    - Cursor
    - Coplit
  - 国产：
    - Qwen Code
    - 通义灵码
    - Comate
    - CodeBuddy
- WEB端产品
  - Bito AI
  - CodeRabbit
  - Sourcery

02

# AI代码审查在B站的实践

# 系统全貌


MR webhook
gitlab
Git clone
add Comment
Generate
Pipeline
add
CI Server1
CI Server2
Agent
Agent
.....
Agent
display
Platform Server
Platform
CodeRAG
RAG
Graph DB
Vector DB
LLM
Issues
Suggestions
DB

# 系统架构


AI CodeReview
交互层
MR内
变更审查总览
MR行间评论
审查平台
问题反馈
历史数据及问题追踪
知识层
工程编码规范
业务自定义规范
内部技术文档
业务缺陷集
框架层
Chat
Agent
RAG
CodeRAG
模型层
GPT
DeepSeek
Claude
Gemini
Qwen
……

# 实践落地中的挑战


## 代码上下文缺失

仅仅关注代码diff部分的简单上下文，能够支撑高质量的审查结果吗?

## 效果评测

模型切换、框架切换、提示词更新，如何对系统进行重新评估?

## 准确性/稳定性

大模型输出结果存在不确定性，直接影响审查效果，如何进行修正?

## 用户感知

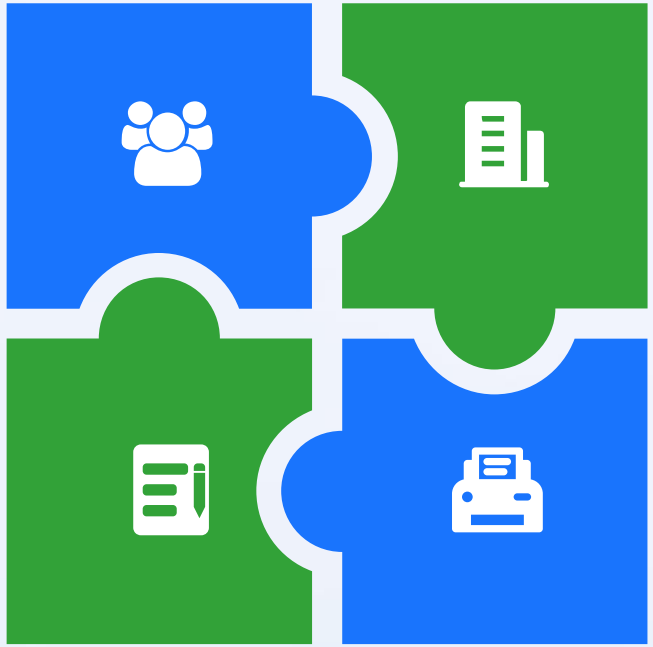
如何让用户感知到整套系统的存在，并深度融合研测流程?

# 挑战1：代码上下文缺失

如何通过变更片段感知整体仓库?

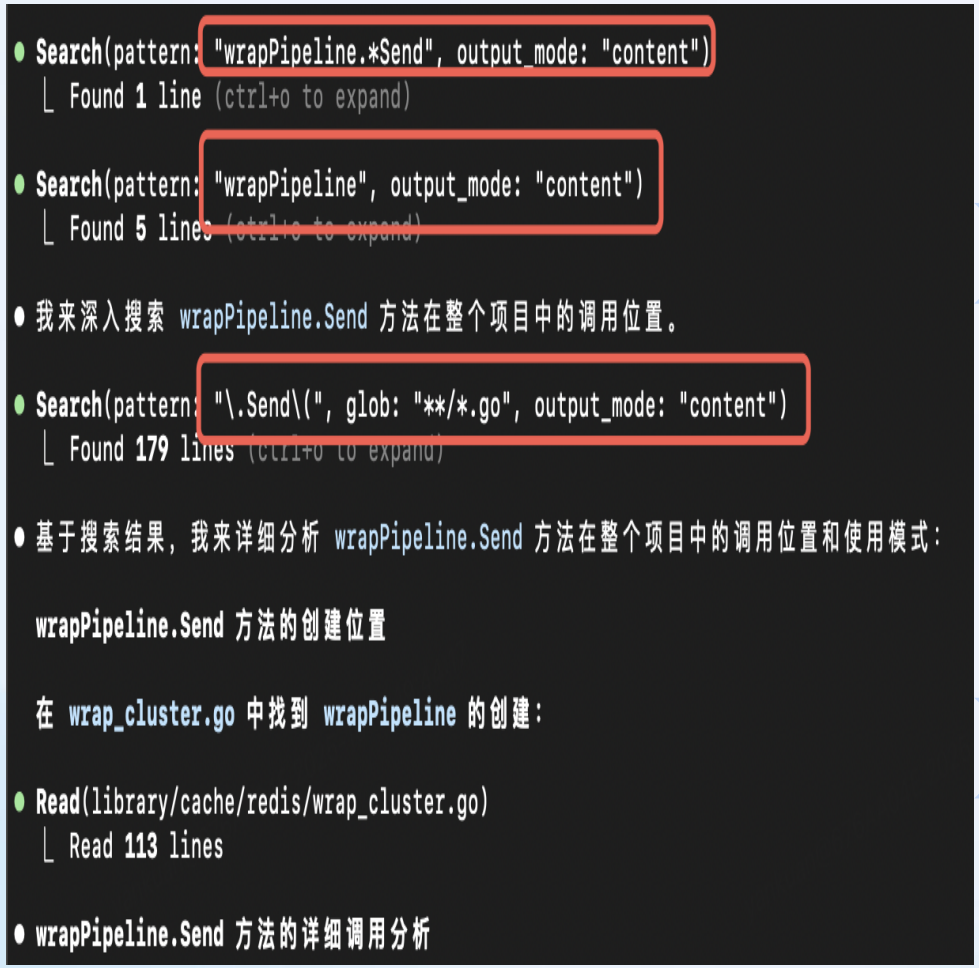
变量、函数、类从哪里来?

调用链路是否被破坏?



跨文件/跨模块调用关系是怎么样的?

# 方案比较：文本检索




1

Glob：根据给定通配符，在文件系统中递归查到文件

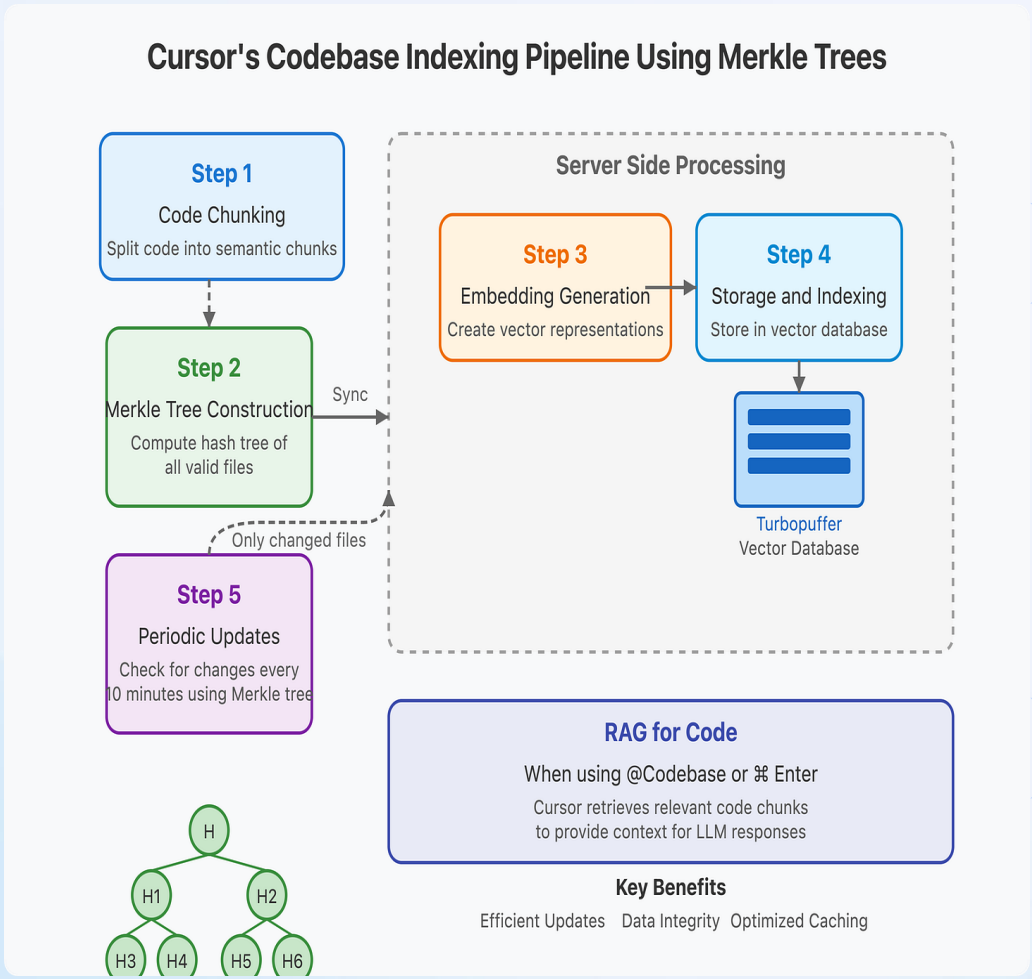
Grep：根据给定正则pattern，在文件中匹配相关内容

2

方案代表：Claude Code/Gemini Cli

方案特点：
实施简单，没有语义，没有结构，只是原始的字符串匹配
检索精度带来的误匹配/漏匹配问题，项目体积膨胀带来的检索成本问题

# 方案比较：RAG+构建索引




1

RAG：相比于传统RAG对于文档类知识的切块方法，使用AST可以最大程度保证代码语义和连续性，在检索方面，仍使用语义相似度的方法

代码索引： Merkle树树形层级结构，节点存储hash值，作为索引底层结构，方便响应代码更新

2

方案代表：Cursor

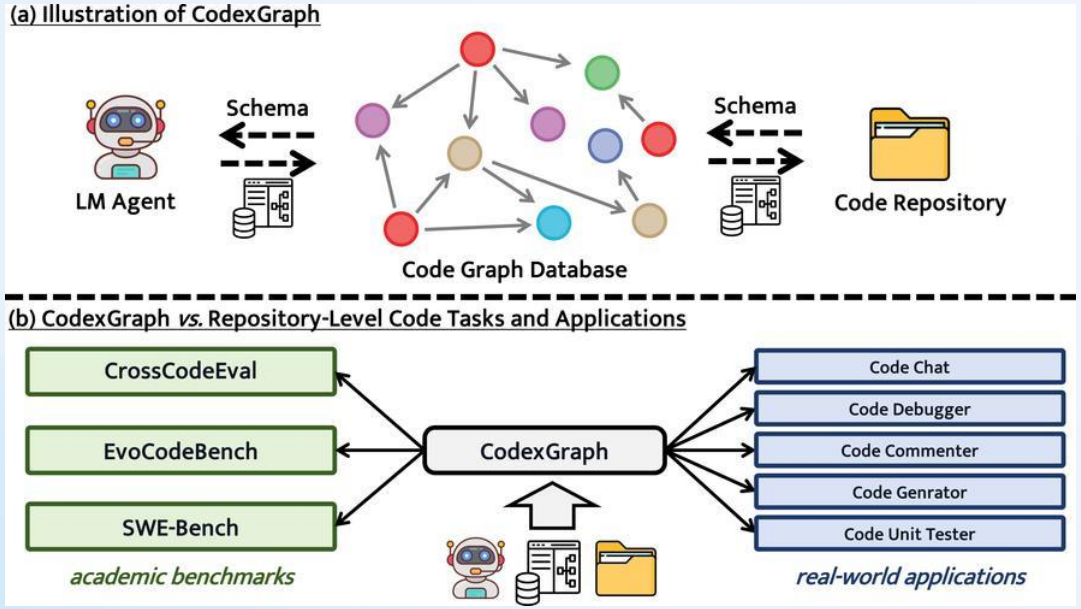
方案特点：
更适合本地IDE场景 ， 而非流水线执行场景
语义相似度进行代码检索，并非代码链路的真实体现

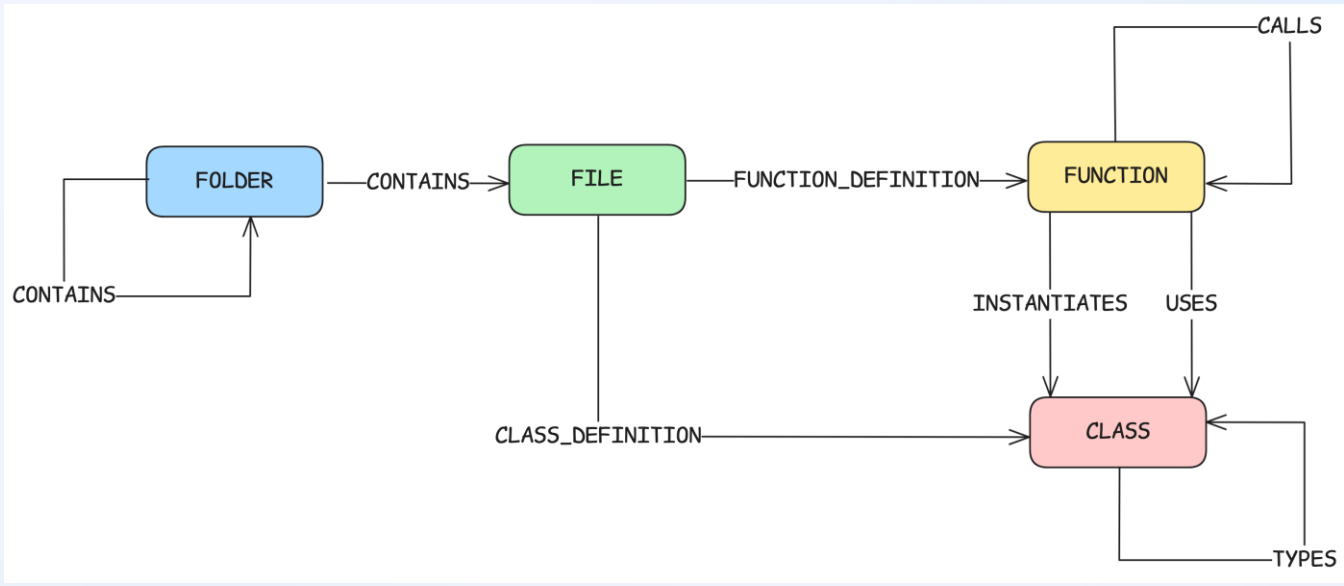
# 解决方案：构建CodeGraph实现CodeRAG

## 让大模型看到真实的代码链路

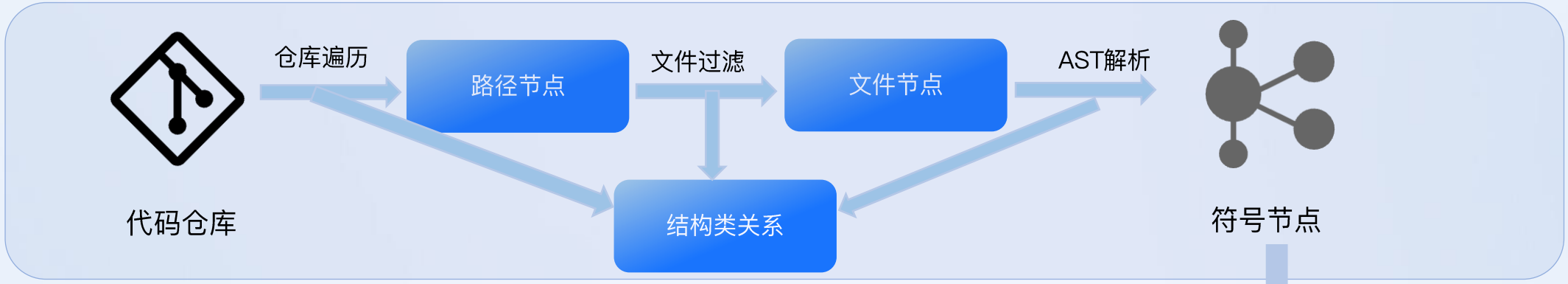
CodeBase –> CodeGraph

三元组定义



# 构建CodeGraph：代码节点解析




```go
func (c *Controller) NormalMergeProcessor(m *mr.TideMrDetail) {
    defer func() {
        if err := recover(); err != nil {
            log.Error( format: "NormalMergeProcessor panic err: %v
        }
    }()
    var (
        org  = m.Org
        repo = m.Repository
        mrID = m.MrID
        pid  = fmt.Sprintf( format: "%v/%v", org, repo)
    )

    if m.State != mr.Merged {
        var (
            res = map[string]bool{
                "noConflict":     false,
                "tapd":           false,
                "reviewed":       false,
                "agileJobPassed": false,
                "notWip":         false,
                "noAgentIssues":  false,
            }
```

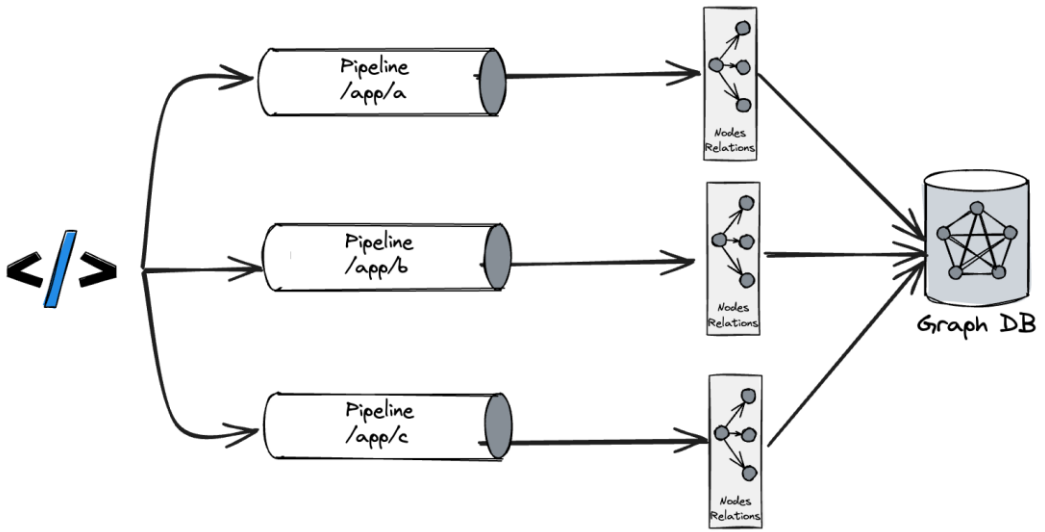

```json
{
    "type": "FUNCTION",
    "extra_labels": [],
    "attributes": {
        "label": "FUNCTION",
        "path": "controller/controller.go",
        "node_id": "da1b42ba6b1dcb26b0e046ed559784ba",
        "node_path": "controller/controller.go.NormalMergeProcessor",
        "name": "NormalMergeProcessor",
        "level": 2,
        "hashed_id": "da1b42ba6b1dcb26b0e046ed559784ba",
        "diff_identifier": "",
        "repo_id": "ep-platform/agileflow-tide/:main",
        "app_id": "test.ep.agileflow-tide",
        "node_ownership": 0,
        "commit_id": "1ecc2eaa",
        "branch": "main",
        "stats_max_indentation": 0,
        "stats_min_indentation": 0,
        "stats_average_indentation": 0,
        "stats_sd_indentation": 0.0,
        "start_line": 189,
        "end_line": 368,
        "text": "func (c *Controller) NormalMergeProcessor(m *mr.TideMrDetail)
        "stats_parameter_count": 1,
        "parameters": "*mr.TideMrDetail",
        "receiver": "*Controller"
    }
},
```

# 构建CodeGraph：语义关系挖掘

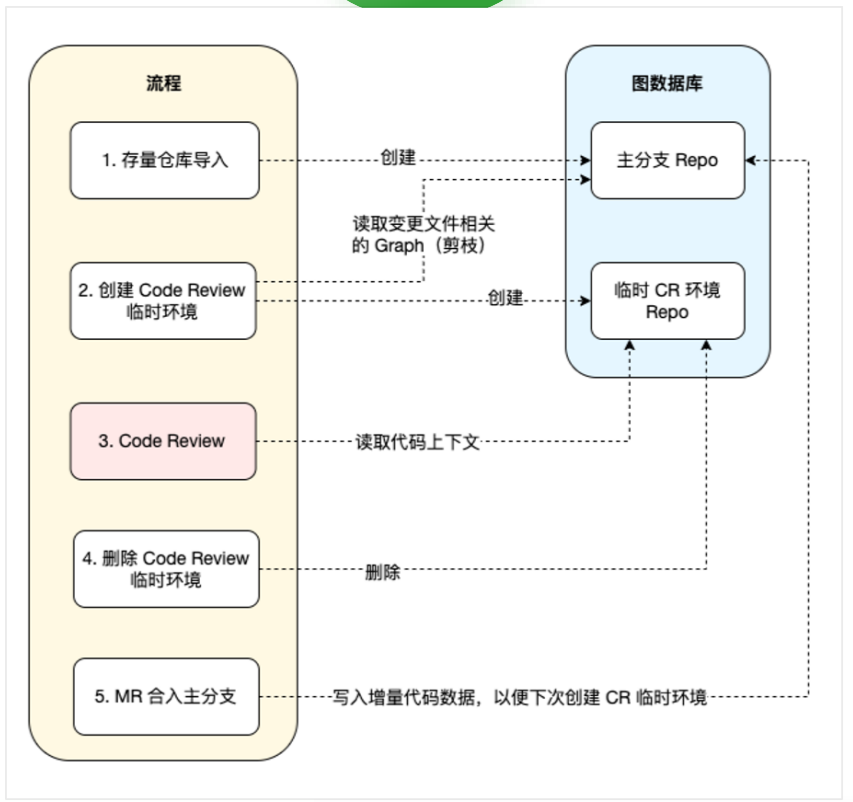

```
190     func (c *Controller) NormalMergeProcessor(m *mr.TideMrDetail) {
229             log.Warn( format: "tide: controller mr may be conflict
230             return
231         } else {
232             res["noConflict"] = true
233         }
234
235         // 检查mr是否处于wip状态
236         notWip := c.checkWip(m)
237         if notWip {
238             res["notWip"] = true
239         } else {
240             log.Warn( format: "tide: controller mr in wip. mr: %v/
241         }
242
243         // 检查tapd是否关联
244         tapdAssociated := c.checkTapd(m)
245         if tapdAssociated {
246             res["tapd"] = true
247         } else {
248             log.Warn( format: "tide: controller mr tapd not associ
249         }
250
```

```
3026     {
3027         "sourceId": "da1b42ba6b1dcb26b0e046ed559784ba",
3028         "targetId": "e5f2467df184a65dbdbf4fb34f78fbb5",
3029         "type": "CALLS",
3030         "scopeText": "c.checkTapd(m)",
3031         "callLine": 244
3032     },
3033     {
3034         "sourceId": "da1b42ba6b1dcb26b0e046ed559784ba",
3035         "targetId": "64f1c095edeac59984fbfafbdaf563e8",
3036         "type": "CALLS",
3037         "scopeText": "c.checkReviewed(m, sha)",
3038         "callLine": 270
3039     },
3040     {
3041         "sourceId": "14f13eb94b066c62d5ddcbc1f9d32612",
3042         "targetId": "2f35c6e0611a3fc2dddc2a7e81bcd999",
3043         "type": "CALLS",
3044         "scopeText": "c.checkMerge(m, false)",
3045         "callLine": 184
3046     },
3047     {
3048         "sourceId": "da1b42ba6b1dcb26b0e046ed559784ba",
3049         "targetId": "2f35c6e0611a3fc2dddc2a7e81bcd999",
3050         "type": "CALLS",
3051         "scopeText": "c.checkMerge(m, isSquash)",
3052         "callLine": 336
```

# 构建CodeGraph：效率提升




MonoRepo拆分




增量构建

# CodeRAG：CodeGraph->CodeContext

Git diff

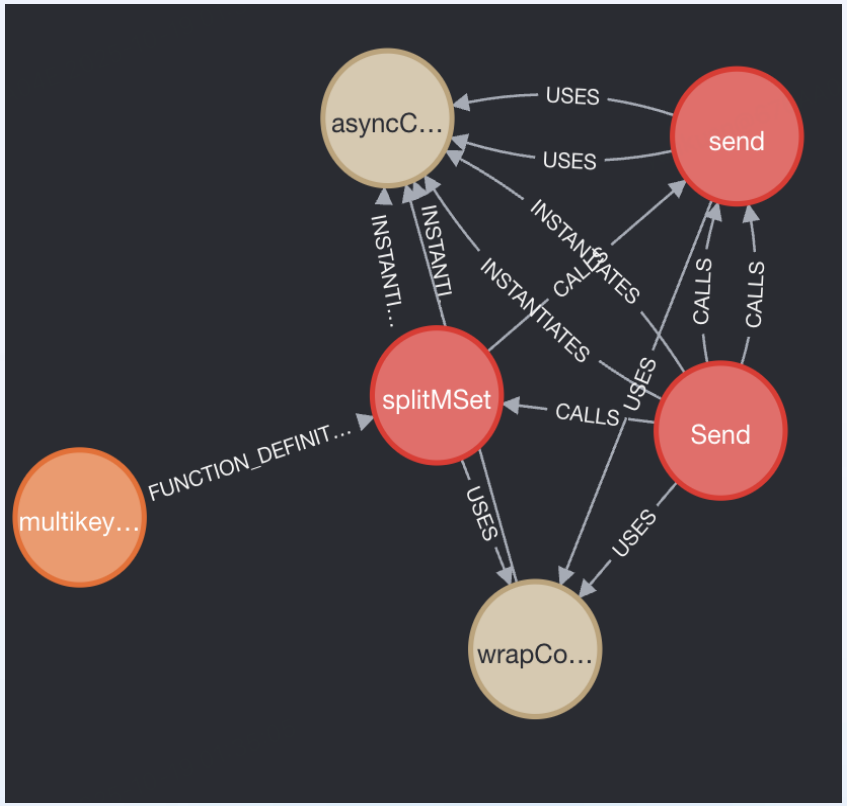
```
112 113   func (c *wrapConn) splitMGet(args ...interface{}) {
113       -         mgetHolder := &asyncCmd{cmdName: "mget", wg: c.wg,
          keyCount: len(args), ctx: c.ctx}
    114   +         mgetHolder := getAsyncCmd(c.ctx, "mget", nil,
          time.Time{}, false, false, 0, len(args), c.wg)
114 115           mgetHolder.multiKeyPlaceHolder =
          mgetHolder.multiKeyPlaceHolder | mgetFlagBit
115 116           c.msgs = append(c.msgs, mgetHolder)
116 117           for _, arg := range args {
117       -                 msg := &asyncCmd{cmdName: "get", keyPos: 0,
          readOnly: true, enableLocalCache: true, args: []interface{}{arg},
          begin: c.begin, wg: c.wg, ctx: c.ctx}
    118   +                 msg := getAsyncCmd(c.ctx, "get", []interface{}
          {arg}, c.begin, true, true, 0, 0, c.wg)
118 119                   c.send(msg)
119 120                   c.msgs = append(c.msgs, msg)
120 121           }
```

AST

```
"library/cache/redis/multikey_cmd.go": {
"classes": [
……
],
"functions": [
……
]
```

Query

CodeGraph




Generate

CodeContext

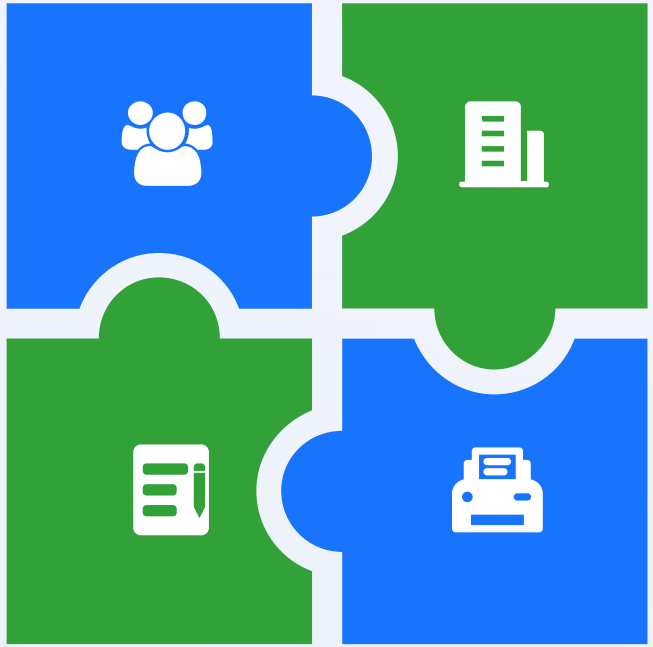
```
# 代码上下文
以下是和 git diff 相关的代码上下文信息，你可以结合这些信息进行分析。
## 函数路径：
library/cache/redis/multikey_cmd.go.splitMSet
……
### 被调用相关函数:
……
### 上层函数（调用该函数的函数）：
……

## 类路径：
……
### 字段相关类型:
……

## 同一个模块下的文件：
……
```

# 挑战2：准确率/稳定性提升

模型给出的行号不准确怎么办?



问题类型太宽泛，无法对重点问题进行召回怎么办?

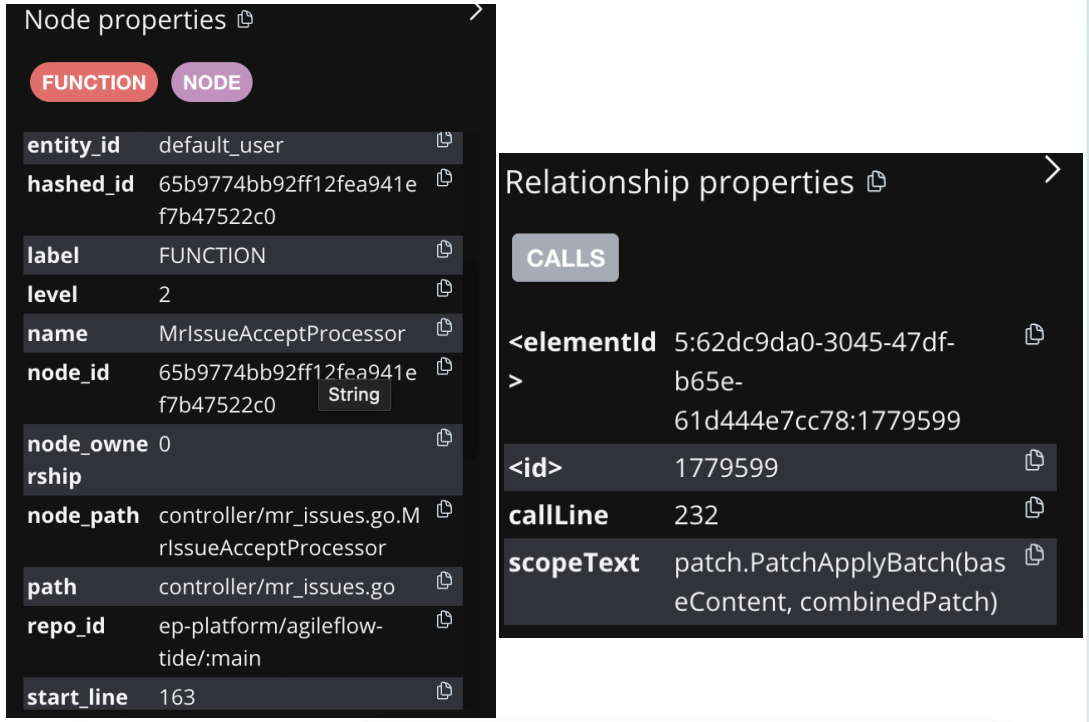
模型无法理解特定业务场景下的问题怎么办?

模型多次审查发现的问题都不一样怎么办?

# 行号修正提升审查准确率

```
# 输入数据
- git diff 内容，其中 - 表示旧版本，+表示新版本，最左侧的数字是对应的文件行号，-表示旧行号，请忽略:
diff --git a/library/queue/databus/options.go b/library/queue/databus/options.go
index fb6a5e1f08e..cb10face360 100644
--- a/library/queue/databus/options.go
+++ b/library/queue/databus/options.go
@@ -82,7 +82,7 @@ func TraceInject(ctx context.Context, topic string, group string) string {
 82      return m[trace.BiliTraceID]
 83  }
 84
-000  func NewContext(ctx context.Context, topic string, group string, traceID string, propert
ntext.Context {
+85  func NewContext(ctx context.Context, properties map[string]string) context.Context {
 86      md := make(metadata.MD, len(properties))
 87      for k, v := range properties {
 88          md[k] = v
```

行号标注




查询CodeGraph

# 共建审查规范提升审查准确率

## 1 工程团队：编码规范

| 规则名 | 是否拦截 | 规则详情 |
| --- | --- | --- |
| 连接风暴 | 是 | - 频繁新建SDK/GRPC连接，造成连接风暴 |
| 队列检查 | 是 | - 队列/通道写入无背压与超时，易阻塞与堆积 |
| 流量分配比例 | 是 | - 降级比例/分母变更导致路由与流量错误分配 |
| 重试放大 | 是 | - 网关/客户端重试在5xx/超时场景放大流量，缺少幂等/退避/上限 |
| 结构体字段变更 | 是 | - 接口响应字段类型/必填属性变更未向后兼容，导致网关/下游解析失败与5xx |
| 日志打印 | 是 | - 日志打印大对象或热循环逐条打印，导致CPU/IO放大 |

## 2 业务研发：自定义规范


## 3 架构团队：内部技术文档


**让大模型了解业务需要的审查规范究竟是什么**

# Multi-Agent提升持续审查稳定性

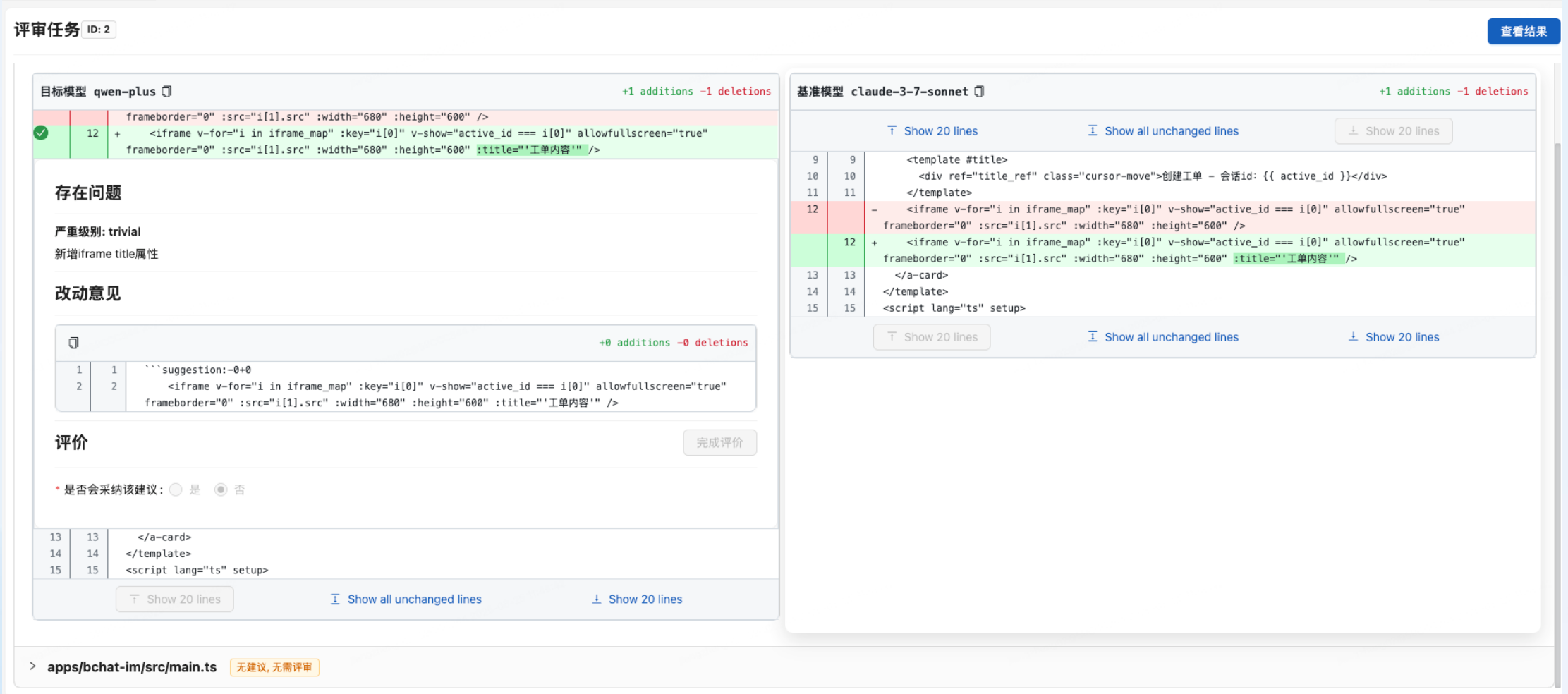
# 挑战3–效果评测：评测标准

| 类别 | 指标 | 描述 | 评价方式 |
| --- | --- | --- | --- |
| 效率类 | 耗时 | 完成一次分析的时间开销 | 记录起止时间 |
| | LLM Token 消耗 | 分析过程中消耗的 Token 数 | 统计 API 调用 Token 使用量 |
| 质量类 | 上下文理解 | 是否准确理解函数调用关系和依赖 | LLM 评估 1–10分 |
| | 问题识别准确性 | 是否准确识别出代码中的潜在问题 | |
| | 修复建议质量 | 修复建议是否合理、符合语言最佳实践 | |
| | 性能考量 | 是否考虑代码变更可能带来的性能影响 | |
| | 安全性评估 | 是否发现潜在安全风险并给出提示 | |
| | 可维护性分析 | 是否评估代码的可读性、可扩展性、可测试性 | |
| | 错误率 | 是否存在错误或误导性建议 | |
| 准确性 | 准确率 | 正确识别的问题 / 总识别出的问题 (Precision) | P = TP / (TP + FP) |
| 覆盖率 | 召回率 | 正确识别的问题 / 实际存在的问题 (Recall) | R = TP / (TP + FN) |

# 效果评测：数据集制作

| 类别 | 生产方式 | 描述 | 正例/反例 |
| --- | --- | --- | --- |
| AI生成问题 | AI生成 | 用于系统初期建设或罕见问题的评测 | 正例 |
| 线上问题反馈 | 人工录入 | 对线上实际发生的问题和问题代码段进行绑定用于评测 | 正例 |
| QA测试BUG | 人工录入 | 对QA测试过程中发现的BUG和问题代码段进行绑定用于评测 | 正例 |
| AI代码审查发现问题 | 自动录入 | 对用户采纳修复意见的问题作为正例用于评测 | 正例 |
| 用户标注误报 | 自动录入 | 将用户任务误报的问题作为反例用于评测 | 反例 |

# 效果评测：评测平台


评审任务 ID: 2
查看结果
目标模型 qwen-plus
+1 additions -1 deletions
frameborder="0" :src="i[1].src" :width="680" :height="600" />
12 + <iframe v-for="i in iframe_map" :key="i[0]" v-show="active_id === i[0]" allowfullscreen="true" frameborder="0" :src="i[1].src" :width="680" :height="600" :title="'工单内容'" />
存在问题
严重级别: trivial
新增iframe title属性
改动意见
+0 additions -0 deletions
1 1 ```suggestion:-0+0
2 2 <iframe v-for="i in iframe_map" :key="i[0]" v-show="active_id === i[0]" allowfullscreen="true" frameborder="0" :src="i[1].src" :width="680" :height="600" :title="'工单内容'" />
评价
完成评价
* 是否会采纳该建议： 是 否
13 13 </a-card>
14 14 </template>
15 15 <script lang="ts" setup>
Show 20 lines
Show all unchanged lines
Show 20 lines
基准模型 claude-3-7-sonnet
+1 additions -1 deletions
Show 20 lines
Show all unchanged lines
Show 20 lines
9 9 <template #title>
10 10 创建工单 - 会话id: {{ active_id }}
11 11 </template>
12 - <iframe v-for="i in iframe_map" :key="i[0]" v-show="active_id === i[0]" allowfullscreen="true" frameborder="0" :src="i[1].src" :width="680" :height="600" />
12 + <iframe v-for="i in iframe_map" :key="i[0]" v-show="active_id === i[0]" allowfullscreen="true" frameborder="0" :src="i[1].src" :width="680" :height="600" :title="'工单内容'" />
13 13 </a-card>
14 14 </template>
15 15 <script lang="ts" setup>
Show 20 lines
Show all unchanged lines
Show 20 lines
apps/bchat-im/src/main.ts
无建议, 无需评审

# 挑战4–用户感知：融合DevOps

打通CI流水线，响应MR中的代码提交，持续对代码进行自动化审查

# 用户感知：代码准入

将AI CodeReview的结果，作为人工代码审查以及流水线执行结果的补充，共同作为MR是否能够合入的标准

# 用户感知：MR变更审查总览

agile @agile · 1 day ago　Maintainer

AI CodeReview 报告

得分：55 | 不建议合并

1. 审查任务进度(401ddde1)

初始化 → 重点问题 → 建议问题 → 总体评价

2. 审查结果

▾ 重点关注问题

| 文件 | 问题描述 | 严重级别 | 增量/存量 | 状态 | 链接 |
|---|---|---|---|---|---|
| library/stat/sys/cpu/cpu.go | refreshCPU函数存在除零风险 | high | 存量 | 待处理 | link |
| library/stat/sys/cpu/cpu.go | 全局变量idx存在数据竞争 | medium | 存量 | 待处理 | link |
| library/stat/sys/cpu/cpu.go | isValidaQuota函数已修复 | low | 存量 | 已解决 | link |
| library/stat/sys/cpu/cpu_linux.go | 访问parts[0]可能越界panic | high | 存量 | 待处理 | link |
| library/stat/sys/cpu/cpu_linux.go | cgQueryer并发访问可能data race | medium | 存量 | 待处理 | link |

▾ 建议观察问题

| 文件 | 问题描述 | 状态 | 链接 |
|---|---|---|---|
| library/stat/sys/cpu/cpu_linux.go | cgQueryer相关变量缺乏注释 | 待处理 | link |
| library/stat/sys/cpu/cpu.go | psCpuUsage中idx存在数据竞争 | 待处理 | link |

3. 总体评价

这次MR主要改进了cgroup查询模块和CPU配额计算逻辑，新增了cgQueryer单例模式实现。代码质量存在严重缺陷，包含多个高危风险未解决：

1. **核心panic风险**：systemCPUUsage函数存在parts[0]越界访问风险（cpu_linux.go第132行），refreshCPU函数存在除零panic风险（cpu.go第143行）；
2. **并发安全缺陷**：idx全局变量存在数据竞争（cpu.go第111/147行），cgQueryer存在潜在data race（cpu_linux.go）；
3. **代码可维护性不足**：cgQueryer相关变量缺乏注释说明。

本次变更修复了isValidaQuota函数的NaN检测逻辑（已解决），但保留了所有高风险问题。建议暂缓合入，需优先修复越界访问、除零异常和并发安全问题。

Edited by agile 1 day ago

## 总览评论：

- 审查任务进度：根据用户配置的审查任务开关，编排审查流程，并展示执行进度
- 审查问题概览：展示审查发现问题的基本信息及状态，并提供跳转行间评论/审查平台的入口
- 变更总评：对本次变更进行整体评价，得到一个变更打分，并对是否应该合入主干进行指导

# 用户感知：MR行间评论


## 行间评论：

- threads：根据问题范围确定行号，将问题以行间评论的形式写入到mr中并发起一个thread，方便研发快速定位问题
- 问题详情：展示审查问题的详细描述及改动意见
- Suggest Change：对于有明确修复路径的问题，生成suggestion格式的修复结果，方便研发自动修复问题

# 用户感知：审查平台

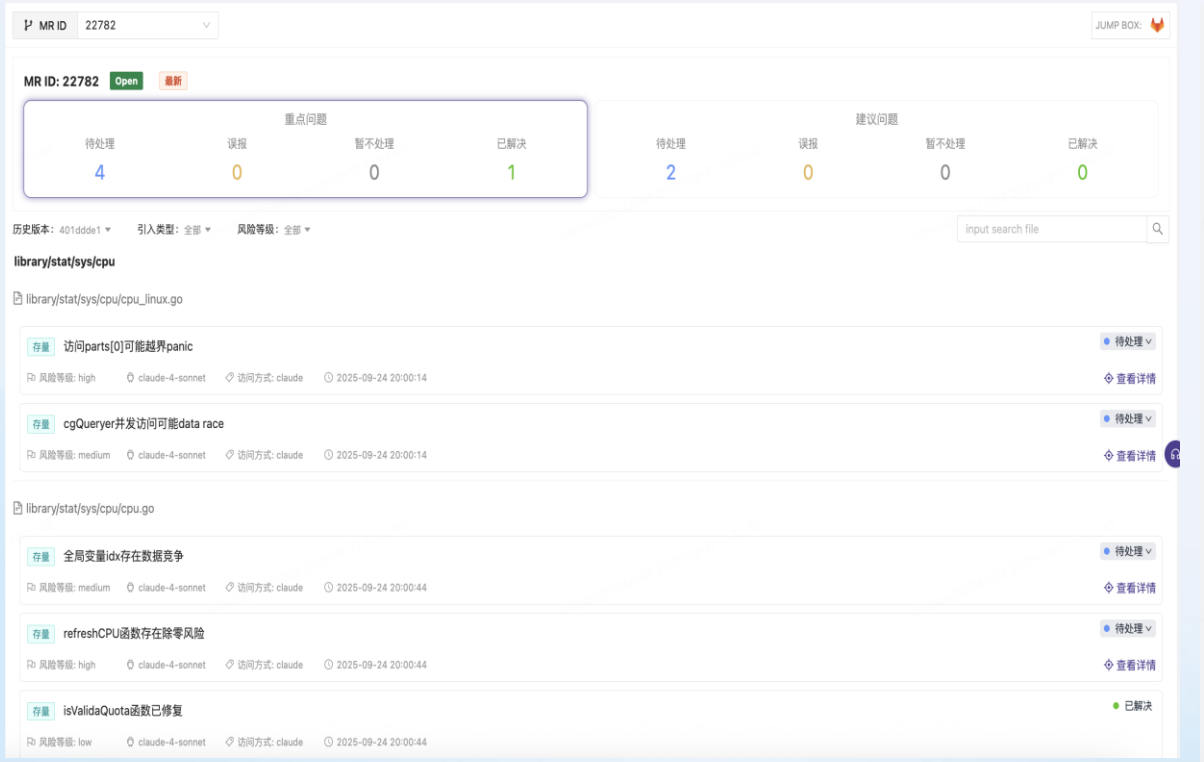
## 数据聚合及追踪：

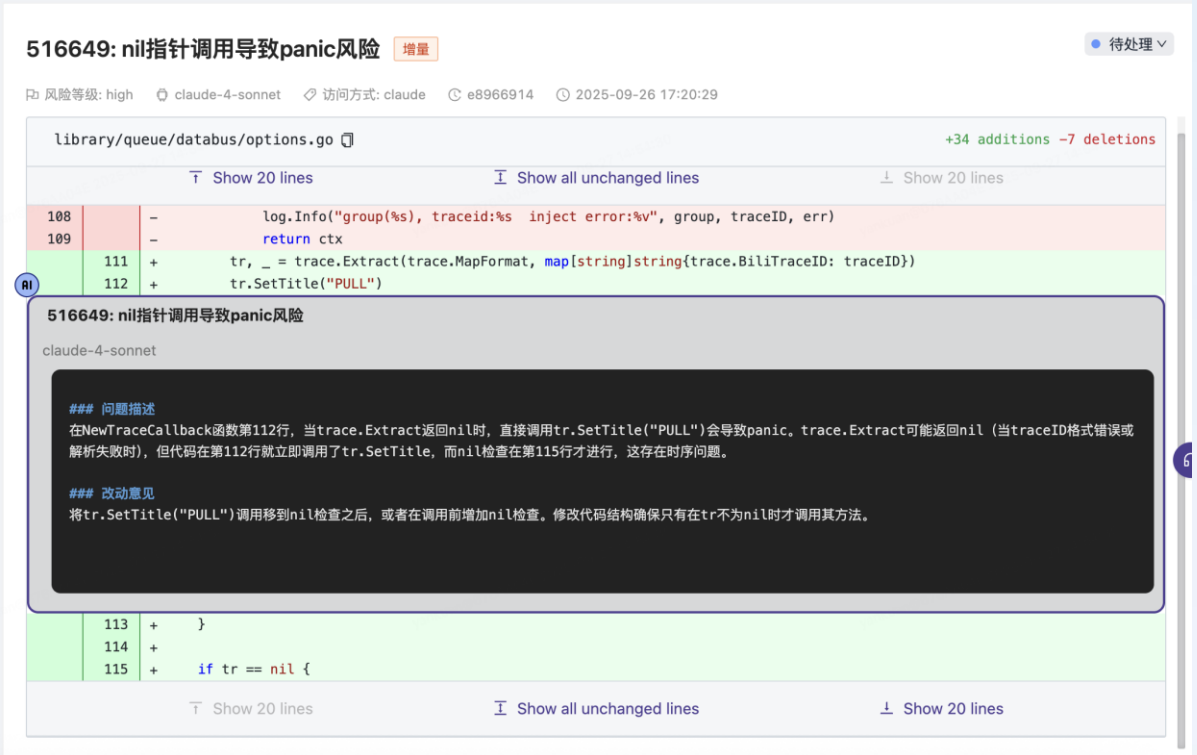
将仓库历史MR发现的问题进行聚合展示

对仓库代码质量健康情况进行追踪及评估

## 问题反馈：

提供反馈入口，对于置为 误报/暂不处理 的问题将不会在后续的审查结果中出现



# 实践效果：典型案例

## 逻辑校验缺失

/option_inspection.go

```
14  27  }
15  28
16  29  func (p *InspectionParam) AppendQuery(...) {
    30  +       if p.subType != "" {
    31  +               q = q.Where(table.MetricSubType.Eq(string(p.subType)))
    32  +       }
    33  +       if p.PageNum != 0 || p.PageSize != 0 {
```

**agile** @agile · 4 months ago　Maintainer

▼AI 代码分析

### 存在问题

**严重级别：major**

1.当前分页逻辑仅检查PageNum和PageSize是否为0，未验证数值有效性，当传入负值时会导致生成错误SQL 2.未对分页参数进行下限校验，可能引发数据库查询异常

### 改动意见

Suggested change　Applied

```
33      -       if p.PageNum != 0 || p.PageSize != 0 {
        33  +   if p.PageNum > 0 && p.PageSize > 0 {
```

## 空指针解引用风险

/guaranteed_select_module.cpp

```
43  44          auto& target_score_info = p_ad->_target_score;
44  45
45  46          auto score = target_score_info._score[target_score_info._score_type];
    47  +       if (p_ad->_splash_schedule_creative_material.first->launch_strategy() == 0) {
```

**agile** @agile · 2 months ago　Maintainer

▼AI 代码分析

### 存在问题

**严重级别：critical**

在修改后的代码片段中，直接访问 p_ad->_splash_schedule_creative_material.first->launch_strategy() 未进行空指针检查。当 _splash_schedule_creative_material.first 为空指针时会导致程序崩溃，影响所有使用该过滤逻辑的请求处理流程，属于高风险的稳定性问题。

### 改动意见

Suggested change　Applied

```
47      -       if (p_ad->_splash_schedule_creative_material.first->launch_strategy() == 0) {
        47  +   if (p_ad->_splash_schedule_creative_material.first != nullptr && p_ad-
            >_splash_schedule_creative_material.first->launch_strategy() == 0) {
```

# 03

# 未来展望

# 未来展望

## 1 审查左移

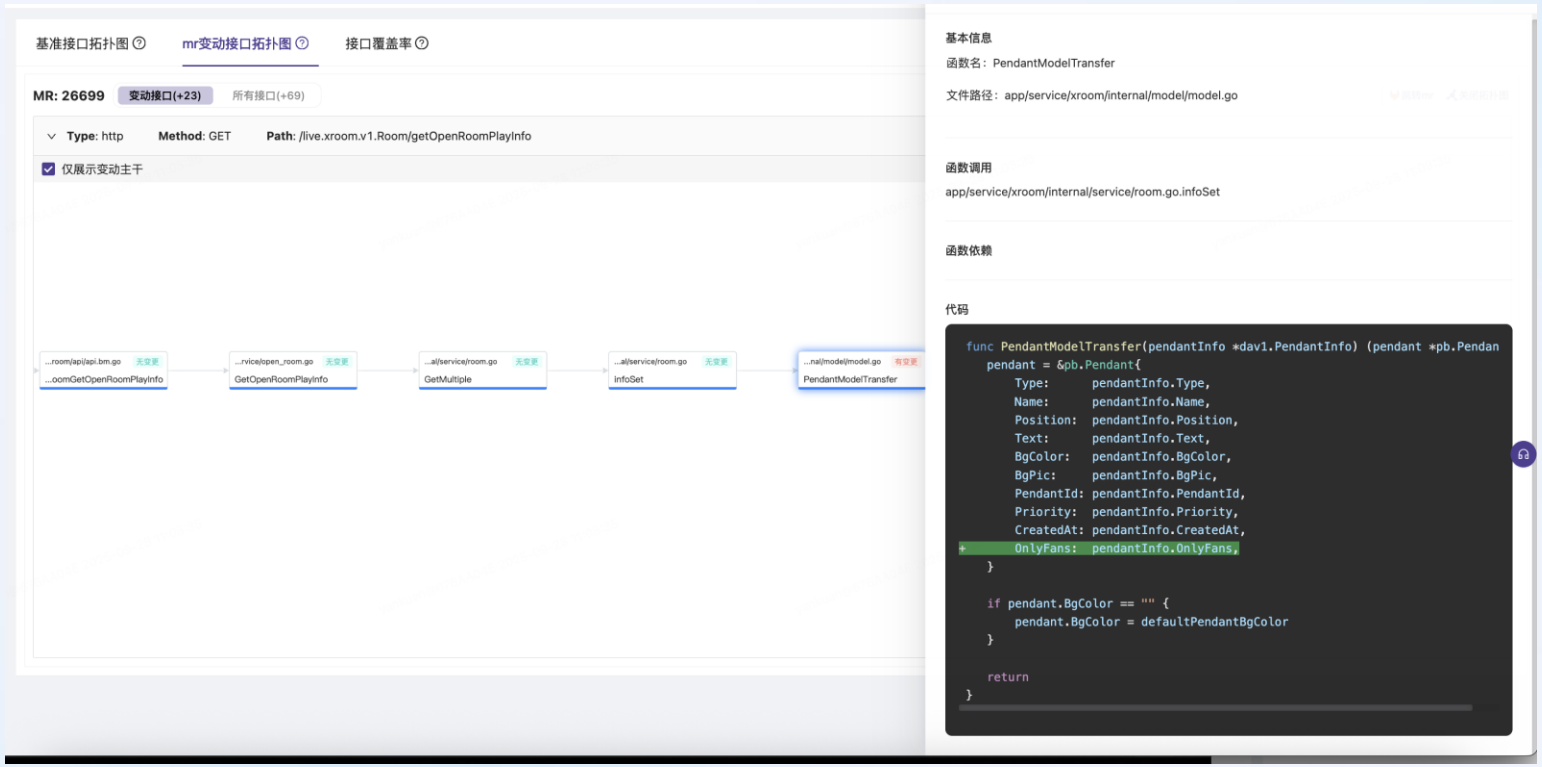
将代码变更阶段的基础上，将能力进行封装，结合IDE插件/CLI工具，在研发本地编码过程中进行审查，提升问题发现及时性，降低修复成本

## 2 需求打通

通过对需求文档进行解析，将其中包含的业务属性作为审查的标准之一，共同对代码变更做出评估

## 3 接口链路打通

将代码变更节点所在的完整接口链路调用信息，以及接口间的调用拓扑信息，共同作为AI评审的私域知识



InfoQ 极客传媒

北京

QCon 全球软件开发大会

会议时间：4月10-12日

- 大模型赋能 AIOps
- 越挫越勇的大前端
- 云上业务架构演进
- 多模态大模型及应用
- 海外 AI 应用创新实践

北京

AiCon 全球人工智能开发与应用大会

会议时间：6月27-28日

- 端侧智能
- AI Agent
- 多模态大模型
- 金融+大模型

上海

QCon 全球软件开发大会

会议时间：10月23-25日

- AI Agent
- 大模型训练推理
- 端侧 AI
- 搜推深度融合
- 数智金融

4月 — 5月 — 6月 — 8月 — 10月 — 12月

上海

AiCon 全球人工智能开发与应用大会

会议时间：5月23-24日

- 通用大模型
- AI Agent
- 垂直领域应用
- 模型可解释性

深圳

AiCon 全球人工智能开发与应用大会

会议时间：8月22-23日

- 模型效率与部署
- 多模态大模型
- 大模型安全
- 智能硬件

北京

AiCon 全球人工智能开发与应用大会

会议时间：12月19-20日

- 通用大模型
- 智能硬件
- LMOps
- 具身智能

# 极客邦科技 2025 年会议规划

促进软件开发及相关领域知识与创新的传播

参会咨询

查看会议

QCon
全球软件开发大会

InfoQ 极客传媒

# THANKS

## 大模型正在重新定义软件

Large Language Model Is Redefining The Software