



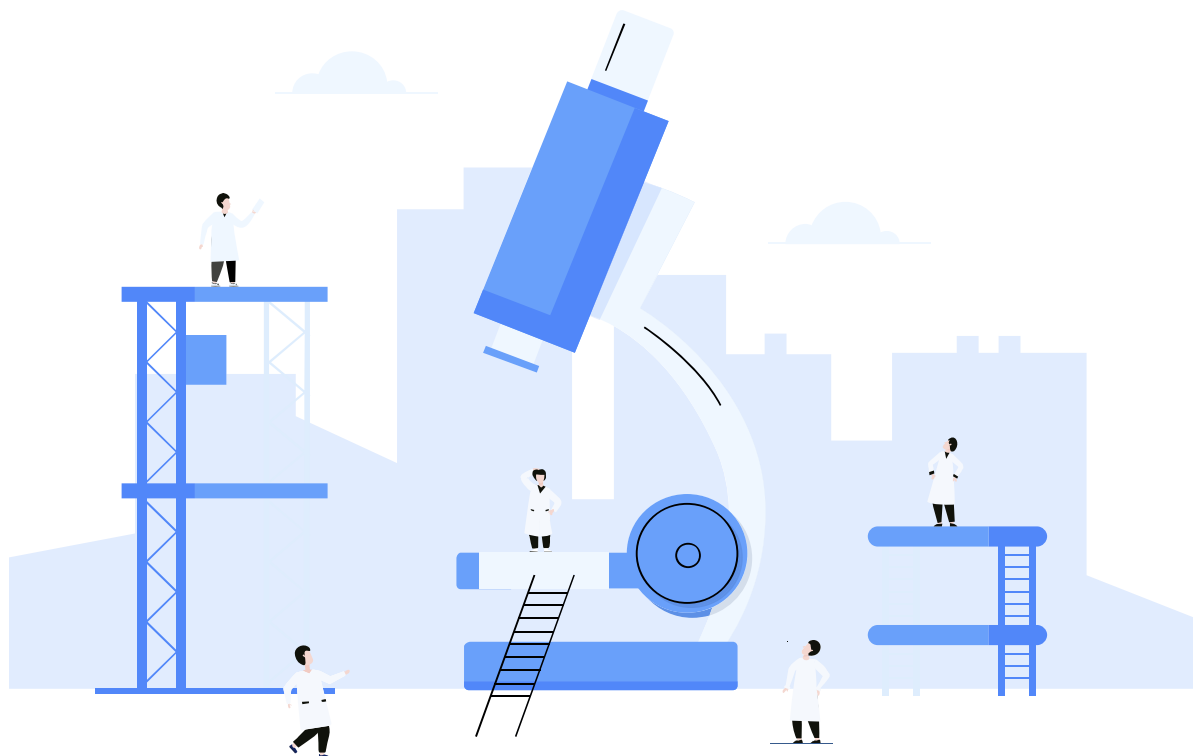
InfoQ 极客传媒

中国企业 研发高效能白皮书

——Code Review 篇



CREATIVE



中国企业 研发高效能白皮书

- 🔊 近年来中国企业研发正在从粗放型走向精益型，研发工作的“高效能”成为几乎每个研发团队共同的追求。
- 🔊 中国软件服务产业也在近5-10年中得到了飞速发展，技术服务的边界不断拓展，赋能研发高效的产品层出不穷，适合中国研发环境的技术服务体系在不断完善。从结果上看，中国企业正在高效能研发的路径上快速前进。
- 🔊 本份报告以中国高效能研发企业为研究对象，尝试解读市场中具有代表性的高效能研发解决方案。本次报告由五个篇章组成，如CI/CD、企业级软件架构、研发效能管理等主题。研究小组期待可以通过研究，帮助中国企业研发团队获得高效能研发新知。



目录

CONTENTS



Code Review 的定义与背景

04

Code Review 的定义

05

Code Review 的价值与挑战

07



Code Review 发展现状

13

Code Review 发展历程

14

Code Review 工具市场的参与者

16

极狐GitLab Code Review 的特征与优势

18



Code Review 最佳实践

23

中海庭

24

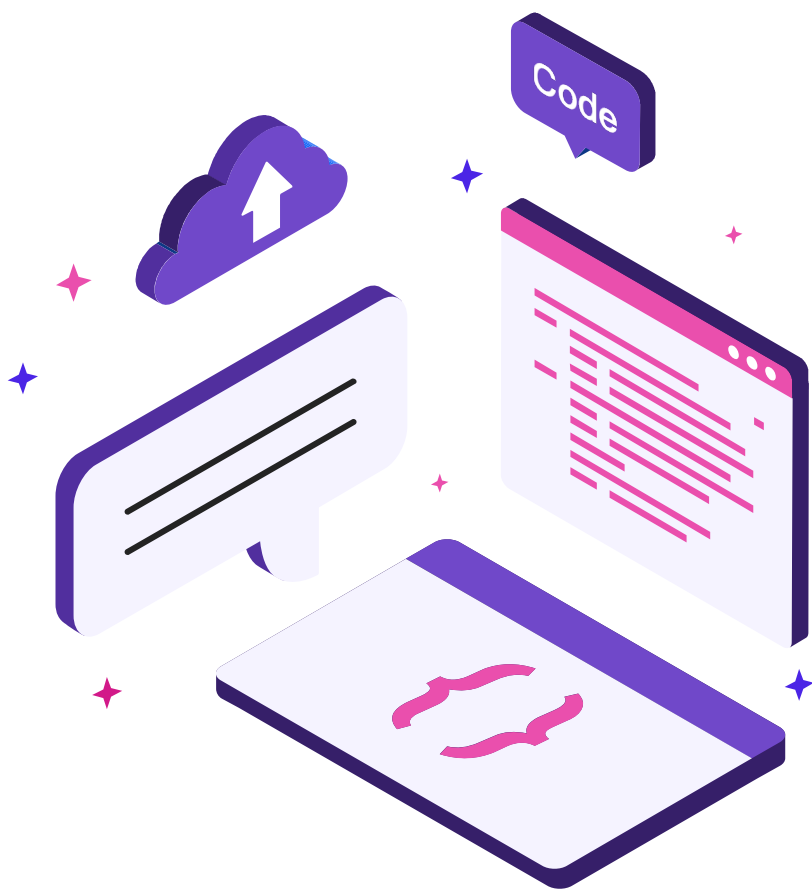
齐碳科技

26

智众医疗

27

Code Review 的定义与背景



Code Review 的定义



Code Review（也称代码评审）是软件开发中对编写的代码进行检查、评审和修改的过程。它是软件开发中一项非常重要的质量保证工作，可以帮助程序员发现潜在的错误、漏洞和缺陷，提高代码质量，降低软件维护成本和运行风险。





“让谷歌的代码如此出色的最重要的因素非常简单：Code Review。并不只有谷歌在做这件事——业界广泛认为 Code Review 是一种非常好的做法，并且很多企业也都在做。但我从未见过其他大公司如此普遍地实行这一做法。在谷歌，任何产品、任何项目的代码，都不会在有效的评审之前被提交到代码库里。”

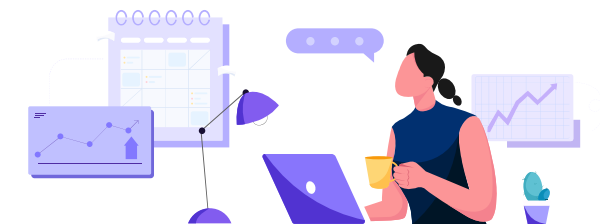
——Mark CC，前谷歌开发者



“Code Review 的好处是显而易见的：有人会仔细检查你的工作是否有错误，反过来，他们也可以从你的解决方案中学到东西。这样的合作有助于提高组织整体的自动化水平。如果你的组织目前没有进行 Code Review，那么现在就开始吧。这会让每个人都成为更好的工程师。”

——Gergely Orosz，优步工程主管

Code Review 的价值与挑战



Code Review 的价值

使用正规
Code Review 的项目,
发现潜在缺陷率约在

60–65%

发现率高

非正规的
Code Review 流程能
发现的潜在缺陷不到

50%

远低于正规 Code Review

Code Review
可以找到
并且移除的错误约

65–85%

移除率高

Code Review 发现
的各种缺陷类型中

50%

与安全隐患有关

降低安全隐患

数据来源:

C. Ebert and C. Jones, "Embedded software: Facts, figures, and future.," IEEE Computer, vol. 42, no. 4, 2009.

Code Review 的价值

01 沉淀高质量代码资产

- 降低后期成本

Code Review 可以在代码合并前，有效地检查出代码漏洞并及时修复。这在很大程度上，可以降低后期修复问题的难度与维护代码仓库的成本。

- 通用代码沉淀

通用算法和数据结构复用，提高整体开发效率。



02 提升团队能力，实现知识流转

Code Review 的一个重要功能就是可以让知识在团队里高效流通。

助力开发者快速成长

开发者可以收到高质量的反馈并及时调整问题，从中快速成长并得到良好的实践体验。

助力开发者融入团队

可以在拉齐团队技术认知的同时，帮助新加入的开发者快速融入团队。

实施 Code Review 的挑战

01 企业面临的挑战

❏ 缺失的 Code Review ❏

Code Review 是一种通过及早发现并解决代码中的潜在问题从而保证代码质量的最佳实践。



在实际研发过程中，由于缺乏自动化工具，或者业务难以忍受人工 Code Review 带来的额外成本，企业可能会选择直接忽略 Code Review 环节，等到问题暴露再进行返工。

❏ 流于形式的 Code Review ❏

这种研发模式下，企业是推崇 Code Review 的，但是缺乏有效的流程将 Code Review 强制执行起来。因此经常会出现：

直接通过：评审者时间不充足或者碍于“面子”，很有可能直接进行合并操作。

自行合并：如果找不到评审者，而开发者有合并权限，则会出现自提自合的情况。

在这种情况下，Code Review 往往流于形式，很难真正实践落地，发挥作用。



实施 Code Review 的挑战

01 企业面临的挑战

≡ 低效的 Code Review ≡

- **缺少数据支撑的 Code Review**

缺乏来自测试、安全扫描、规范扫描等报告数据支撑的 Code Review，容易耗费评审者较长的时间，既难以满足业务对于 Code Review 的时间要求，同时也增加了安全风险。

- **单一评审者的 Code Review**

虽然部分企业建立了 Code Review 环节，并且需要评审者进行评审以后代码才可以合并，但是评审环节只能指定单一评审者，无法同时指定多个评审者。这导致对于一些重要的代码，单一评审者模式很难对变更代码进行多方位和全面的评审，可能会漏掉一些代码缺陷。

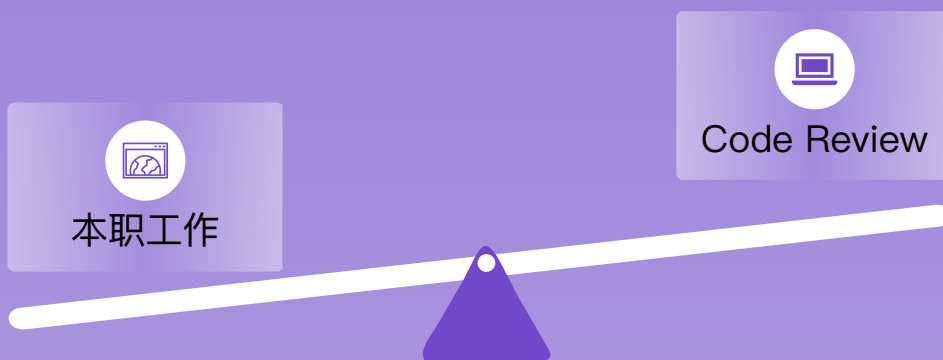


实施 Code Review 的挑战

02 评审者面临的挑战

工作难以平衡

Code Review 常常需要评审者花费额外时间。如果时间不够充裕，就会导致评审者 Code Review 不够彻底或敷衍处理。



如何高效沟通

在进行 Code Review 时，如果评审者与开发者的沟通方式不当，可能会对同事之间的关系产生负面影响。因此，评审者需要尽量展现善意并避免负面词汇的使用，以确保能够有效地提供有建设性的反馈，同时保持良好的团队合作氛围。



实施 Code Review 的挑战

03 开发者面临的挑战

• 选择合适的评审者

Code Review 经常涉及协同更高级别的开发者。这对于一些经验不够丰富的开发者而言，如何选择合适的评审者往往会成为整个评审流程能否顺利推进的重要原因。

01

• 如何推动及时评审

当需要将评审发送给更高级别的多位开发者时，更低级别的开发者经常面临无法按时推进流程的挑战。因此，如何有效地推动评审流程成为大部分初级开发者面临的重要挑战。

02

• 理解并接受建议

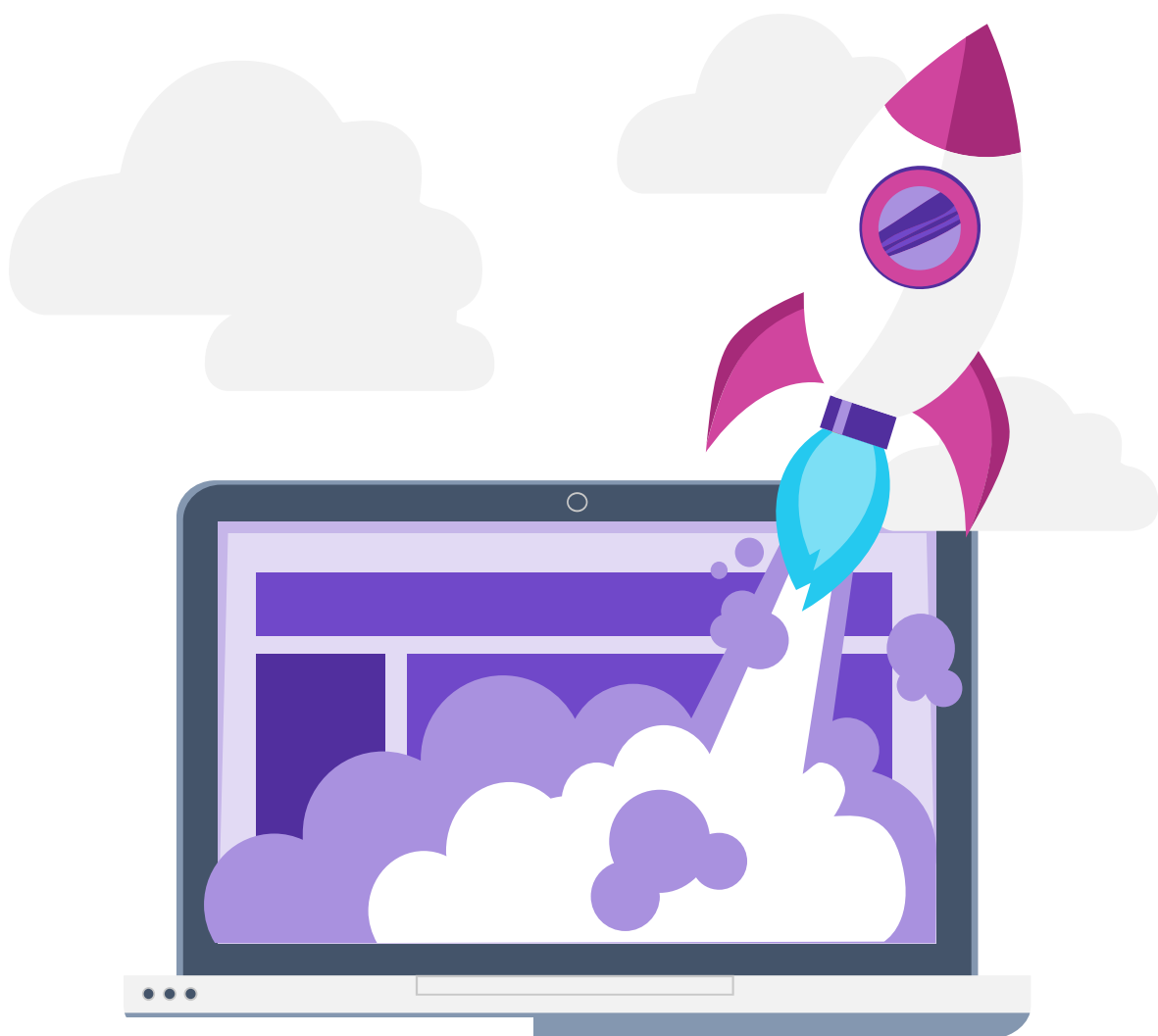
评审者和开发者的技术背景及经验可能不同，这可能导致双方在沟通和问题理解上存在障碍。在这种情况下，开发者需要尽可能清晰地表达自身的想法，并充分理解评审者的意见和建议。

03



Code Review

发展现状



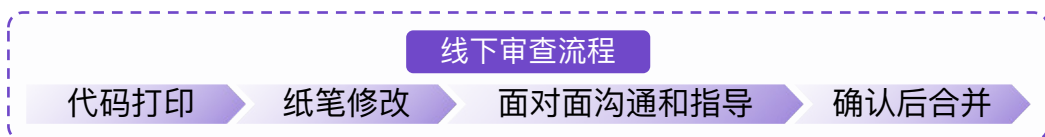
Code Review 发展历程



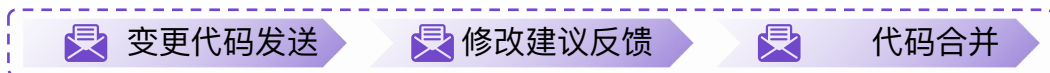
在 Code Review 的整体发展过程中，企业对 Code Review 的要求在逐渐增高，Code Review 工具也在使用需求变化与技术迭代下，不断改进和升级。

阶段一：初始

- ◆ **开始于线下审查：**Code Review 与编程相伴而生。Code Review 最初的形式为线下审查或结对编程。

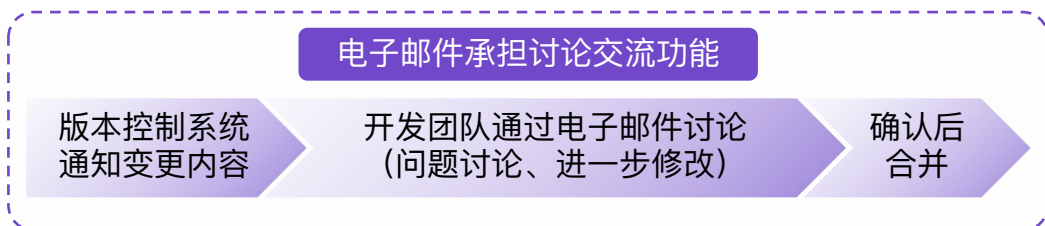


- ◆ 虽然效率相比于现在低很多，但一些企业仍在积极推广，足以看出 Code Review 的重要性。
- ◆ **电子邮件开始承担代码传输功能：**随着电子邮件的兴起，邮件开始承担起代码传输的功能。



阶段二：数字化

- ◆ **Code Review 工具出现：**在2000年左右，随着正式 Code Review 工具的出现，邮件的使用频率逐渐下降，并逐渐不再承担代码传输功能，而是更多地用于公告和讨论。这个时期的 Code Review 工具通常包括代码静态分析、代码风格检查、代码重构建议等功能。



Code Review 发展历程

阶段三：集成化&自动化

- ◆ 2006年开始，Code Review 工具逐渐出现集成化与自动化的趋势，对于 Code Review 的关注度也有大幅地增加。
- ◆ **Code Review 前移**：Code Review 与开发环境高度集成，以便在变更代码合并到主代码库之前启动 Code Review。这可以减少大量重复性劳动，并且帮助评审者更专注于人工检查更为关键的地方。

要求减少重复性劳动：

- 确认构建是否成功
- 确认是否通过了自动化测试
- 确认编写风格是否符合团队

专注于人工检查更为关键的内容：

- 业务逻辑是否合理
- 功能是否正确实现
- 代码是否足够可读与可扩展

- ◆ **电子邮件开始承担代码传输功能**：随着电子邮件的兴起，邮件开始承担起代码传输的功能。

阶段四：智能化

- ◆ **Code Review 工具出现**：2021年以后，随着人工智能技术的积累与爆发，工具开始集成机器学习和自然语言处理等人工智能技术，来提高 Code Review 的准确性和效率。
- ◆ **智能 Code Review 工具兴起**：智能化的 Code Review 工具市场也会随软件开发的普及和需求的增长而扩大。这些工具将越来越受到开发者和企业的青睐，并成为软件开发过程中不可或缺的一部分。



Code Review 工具

市场的参与者

Code Review 工具主要分为独立Code Review 工具和一站式Code Review 工具。

以 Gerrit 、Crucible 为代表的独立 Code Review 工具

以 极狐GitLab、GitHub 为代表的一站式 Code Review 工具

独立 Code Review 工具



Crucible



Review Board



Review Assistant

Barkeep



Collaborator



HelixSwarm



CODACY



CodeFactor



RhodeCode

VERACODE



JArchitect

codestriker

一站式 Code Review 工具



极狐
GITLAB



Azure DevOps



ATLASSIAN
Bitbucket

独立 Code Review 工具



- ◆ 以 Gerrit 为代表的独立 Code Review 工具，在传统的源码管理协作流程中引入了 Code Review 机制：通过人工评审和自动化验证，将不符合要求的代码屏蔽在代码库之外。此外，还可以通过多人校验、多人互备和自动化构建核验来进一步保障核心代码的质量和可靠性。
- ◆ 由于独立 Code Review 工具不具备完整的开发集成生态，在开发的便利度上相对于一站式 Code Review 工具而言有所不足。

一站式 Code Review 工具

01 高功能灵活度和定制化程度

02 内置了丰富的代码扫描工具

03 相关问题直接展示

例如，极狐GitLab 还可以将扫描出的问题在合并请求的界面上直接展示出来，让开发者先自行修复，这有助于提高评审效率并树立开发者信心。

通过使用一站式
Code Review 工具：

数千
小时

工程师工作时间节省

20%

代码质量感知提升

30%

Code Review 过程优化

60%

Code Review 时间节省

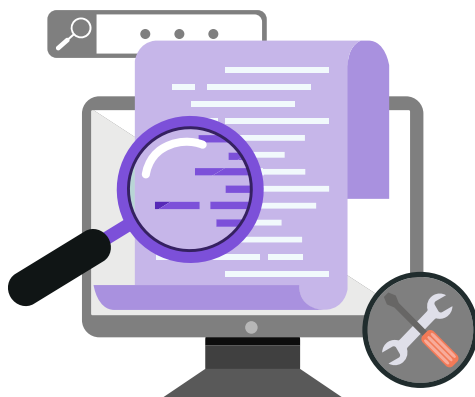
极狐GitLab Code Review 的特征与优势

01

深度嵌入工作流：标准化研发流程，代码工作质量前置

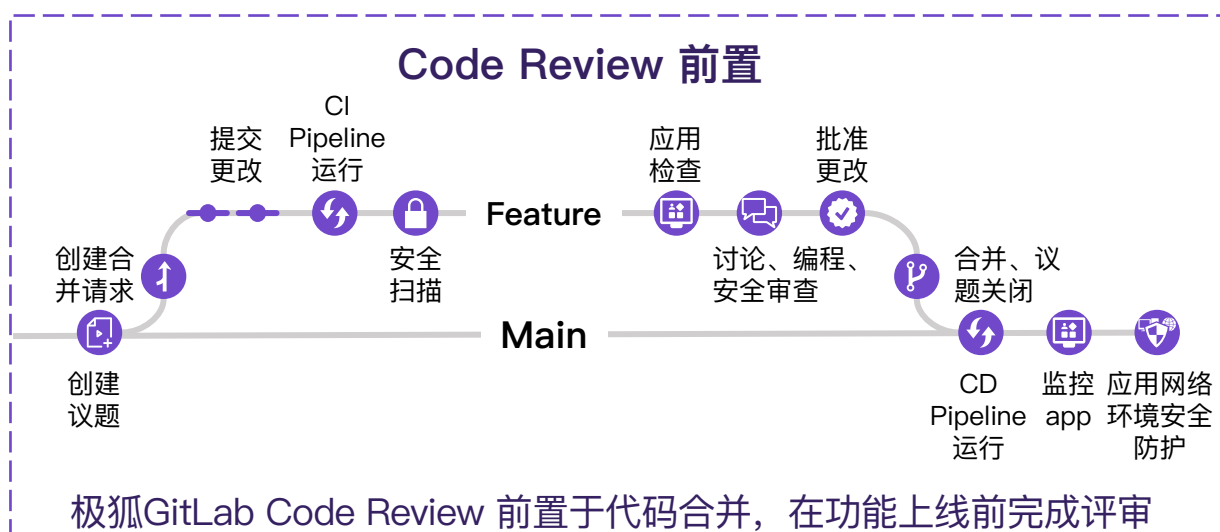
02

高度定制化评审：更专注，更高效



深度嵌入 workflow：标准化研发流程，代码工作质量前置

极狐GitLab 一体化 DevOps 平台，将 Code Review 融入到 JiHu Flow 中，不仅能够大大减轻评审者的工作量，还能够缩短整个 Code Review 周期，高质量完成企业代码质量内建，并且提升研发效率，同时通过硬性规定实现 Code Review，助力企业形成 Code Review 文化。

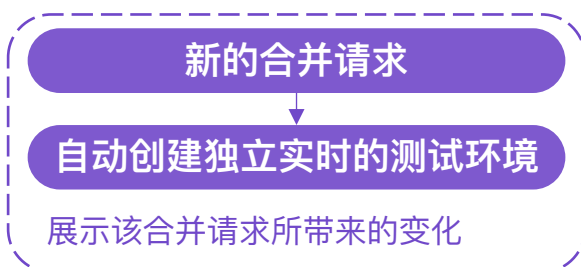


保证安全，提升效率

一体化 DevOps 平台能够让代码变更、CI/CD、Code Review 和合并等操作无需在多个工具之间来回切换。

- 01 避免了数据孤岛的问题。
- 02 有效降低了因认证授权等带来的安全风险。
- 03 保证安全的前提下提高工作效率，让开发过程更加高效和便捷。

适配不同人员需求



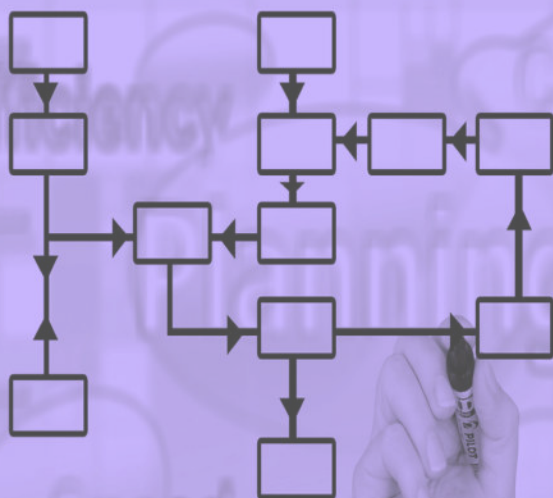
- **开发者**
在可视化、可测试的环境里发现存在的缺陷并及时修复
- **评审者**
减轻评审者的工作量

高度定制化评审

——更专注，更高效

规则多样

- 灵活的评审人数
- 定制准入规则



多维度数据支撑

- 提供增量代码质量报告
- 提供代码单元测试覆盖率总结报告
- 提供开源漏洞库



规则多样

极狐GitLab 支持自定义的评审规则，可以在 UI 界面完成：

01

灵活的评审人数

- **评审者灵活配置机制：**通过批准规则，为代码更改指定评审者和设置最低评审者数量，实现评审者的灵活设置，制定更符合企业自身需求的 Code Review 流程。
- **多人评审机制：**每次代码变更都需要多人评审，这有效地解决了单一评审者无法高质量完成 Code Review 的困境。
- **评审任务移交机制：**如果所选择的评审者没有时间进行评审，也可以把代码的评审权移交给其他评审者。



02

定制准入规则

通过设置代码准入规则，实现 Code Review 的强制执行与高效落地。不同规则的组合设置，提高了代码合并的门槛，帮助企业在研发阶段就更好地落地企业质量内建。

比如：

- **自定义准入规则：**只有通过评审者审批，才能将代码合入主干分支。
- **禁止自行合并代码规则：**设置禁止开发者自行合并代码。



多维度数据支撑

极狐GitLab Code Review 能够提供多样的指标表单，来辅助评审者提高 Code Review 的效率。

每次代码变更都会触发 CI/CD Pipeline 的构建，并生成相关报告（内容包括规范扫描、安全扫描、测试结果等）。评审者可以通过合并请求界面上显示的报告内容，实现以数据为支撑、高效率的 Code Review。

01 提供增量代码质量报告

对变更代码和遗留代码进行分类展现：

- **变更代码**：针对变更代码进行问题挖掘，获得增量代码质量报告。
- **遗留代码**：遗留代码只有在修改之后才会在数据上呈现。

03 提供安全漏洞扫描

- **内置安全扫描功能**：极狐GitLab 内置多种安全扫描功能，可以针对变更代码进行多种安全扫描。

极狐GitLab 内置的安全扫描功能

SAST	DAST
Secret Detection	Dependency Scanning
Fuzz Testing	Container Scanning
License Compliance	

- **安全合并批准规则**：可以要求在合并前必须获得来自安全团队的批准，如果没有获得批准，则无法合并。

02 提供代码单元测试覆盖率总结报告

- **借助报告评估代码的质量和稳定性**：单元测试覆盖率是反映代码质量的重要指标。报告能够提供覆盖率数值并且区分出覆盖与未覆盖部分，评审者可以根据单元测试覆盖率的数据评估代码的质量和稳定性，并针对不足之处提出改进意见。
- **设置测试覆盖率最低通过率**：对于变更代码而言，项目管理者可以在 CI Pipeline 里设置覆盖率门禁，例如通过 Java 插件 JaCoCo，强制拦截测试覆盖率低于设定值的代码。
- **设置覆盖率下降代码拦截**：对于遗留项目的代码，可以通过拦截覆盖率下降的代码来完成代码质量把控。

Code Review

最佳实践





中海庭



使用前

- 源代码管理面临两大难题：

01

原有管理软件无法满足需求

原有管理软件在分支、版本管理方面仍有提升空间

02

代码提交和管理存在风险

需要成熟的工具规避诸如代码被覆盖、误删除等“代码刺客”事件

- 以上难题让高度重视产品质量的中海庭如鲠在喉。后来中海庭切向了 Git，并选择了极狐GitLab CE 版本。为了获得更加专业和贴合本土企业应用场景的服务，中海庭进一步选择了极狐GitLab 企业版本，并在全部产品线推广使用。



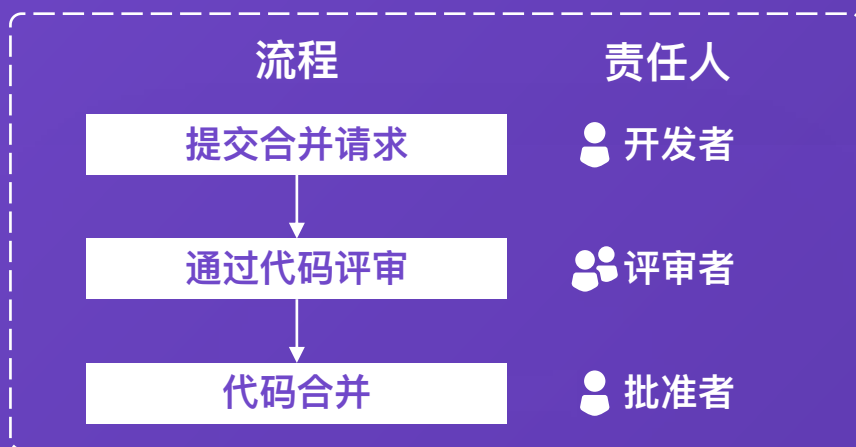


中海庭



使用后

- 借助极狐GitLab 企业版本，中海庭建立了完善的 Code Review 机制：代码需要经过评审和验证才能进入主干分支。



- Code Review 机制达成效果：

01

促进了中海庭研发流程标准化

03

降低了随机事件的发生概率

02

提升了研发效率与产品质量

04

提升了数据制作效率

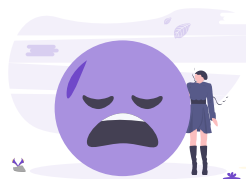
齐碳科技

使用前

每次进行 Code Review 都需要人工沟通寻找合适的评审者：

低效率的团队协同，无法
真正实现 Code Review 01

打击工程师进行 Code
Review 的积极性 02



使用后

齐碳科技形成了**代码多级评审机制**：

- **评审者评审机制**：可以在合并请求中指定多位评审者来对变更进行评审。
- **清晰的评审者任务展现**：每位评审者都可以在极狐GitLab 的 To Do List 上看到需要自己评审的代码。
- **完善的代码合并流程**：在 Code Review 之后，会有对应的批准者进行代码合并。



智众医疗



极狐GitLab 的 Code Review 机制能够保证代码的有效性、可读性、易维护性以及安全性。

业界 Code Review 的通用做法是：需要两位及以上评审者的评审才能进行代码合入。

过去，智众医疗受限于缺少相应的产品，无法指定多个评审者进行 Code Review，不能够充分保证代码质量。

购买极狐GitLab 企业版之后，**进一步保证了提交代码的质量**：

- 可以同时指定**多个评审者**。
- 可以指定**特定评审者**。
- 开发者能够根据评审进行整体修改，最终达到可以合并标准。



敬请期待 更多企业研发高效能白皮书篇章

