

# 源于社区 服务社区

 中国DevOps社区峰会 2023 · 广州



## 从右往左混沌考验：“意外事件助推避坑”与混沌工程

吾真本 - 独立软件开发咨询师





# 吾真本

## 独立软件开发咨询师

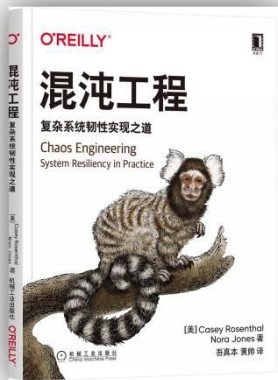
本名伍斌

很多人35岁不写代码，我53岁仍在享受写代码，测试，咨询。



# 吾真本

## 独立软件开发咨询师



2021年开始涉足混沌工程咨询，曾为某金融公司的运维部门，成功交付混沌工程咨询服务，成为Thoughtworks公司中国区首批交付混沌工程咨询项目的咨询师；

“吾真本说混沌工程” 知乎专栏主；





# 目录

- 1 当前生产事故复盘实践的三大缺点
- 2 三大缺点给做软件的人所带来的三大痛点
- 3 “意外事件助推避坑” 四大价值点
- 4 什么是“意外事件助推避坑”？
- 5 如何实现“意外事件助推避坑”？





作为看到IT系统生产意外事件，就有如何助推避坑冲动的混沌工程迷，  
我这3年研究了不少“意外事件助推避坑”案例，  
今天就和大家聊聊我这几年“意外事件助推避坑”与混沌工程实践的经验，  
干货如下：

当前客户端版本 3.2.2  
加载失败，请检查网络状态 [常见问题](#)

重启客户端

```
webContents: did-fail-load  
eventName: undefined  
errorCode: -310  
errorDescription: ERR_TOO_MANY_REDIRECTS  
validatedURL: https://www.yuque.com/500?real_status=500
```

2023.10.23语雀意外事件时用户界面





## 当前生产事故复盘 实践的三大缺点





# 当前生产事故复盘实践的三大缺点

- 1 复盘报告分享难：**生产意外事件复盘报告一般仅供IT团队相关少数几人阅读，使得企业大部分做软件的人不了解以往生产意外事件所揭示的避坑要点，增大了重蹈覆辙的风险，导致再次踩坑；
- 2 演练考试非考验：**仅针对运维部门感兴趣的基础设施层的意外事件（如CPU/内存/磁盘/进程/服务/主机等事件）进行演练，而忽视因程序bug所导致的意外事件演练。当在现实中真正遇到后者，由于缺乏演练，忙乱中难以做出正确判断和行动；
- 3 缺乏亲身体会感：**仅阅读复盘报告，难以亲身体会意外事件发生场景，容易忽视之前所发现过的陷阱。





## 三大缺点给做软件的人所带来的三大痛点







# 三大缺点给做软件的人所带来的三大痛点

## 1 难被认可

“不出问题，不懂的人不知道你做了什么。”

“出了问题，就生气指责你到底做了什么。”（复盘报告分享难）；

## 2 心里发虚

意外事件抢修中由于缺乏演练心发虚（演练考试非考验）；

## 3 很快遗忘

读完意外事件复盘没有实操很快就忘（缺乏亲身体验感）。





## “意外事件助推避坑”四大价值点





# “意外事件助推避坑”四大价值点

## 1 领导更认可

将避坑指南写得利他且易懂以提升稳定性保障领导认可度；

## 2 更多人学到

将生产事故复盘改名为避坑指南以方便扩大分享学习范围；

## 3 练过心不慌

练过各方重视的停机长和损失大的意外事件后遇事则不慌；

## 4 印象更深刻

避坑指南与定期演练相配套以让已发现陷阱印象生动深刻。





# 什么是“意外事件 助推避坑”？



# 什么是“意外事件助推避坑”？

## 什么是“事件”？

指导致系统不可用或离线的特定事情的发生。包括计划外事件和计划内事件。  
意外事件指计划外事件。

## 什么是“助推”？

营造容许自由选择的温和环境，引导人们从意外事件中体验到避坑要点，从而印象深刻，有助于避免重蹈覆辙。比如将“生产事故复盘”改名为“意外事件避坑指南”以方便扩大分享学习范围。





# 如何实现“意外事件 助推避坑”？



# “意外事件助推避坑”的原则

预防虽重要，难免遇意外。

修复意外时，系统仍可用。

不去指责人，实操印象深。

利他且易懂，领导更认可。



业界普遍缺乏“可隔离”、“可替换”和“可避坑”这三种有助快速修复的思维

改进措施：

通过这次故障我们深刻认识到，语雀作为一款服务千万级客户的文档产品，应该做到更完善的技术风险保障和高可用架构设计，尤其是面向技术变更操作的“**可监控，可灰度，可回滚**”的系统化建设和流程审计，从同 Region 多副本容灾升级为两地三中心的高可用能力，设计足够的数据和系统冗余实现快速恢复，并进行定期的容灾应急演练。只有这样，才能提升严重基础设施故障时的恢复速度，并从根本上避免这类故障再次出现。为此我们制定了如下改进措施：





# 2023.10.23语雀意外事件简介

起止时间：2023.10.23 14:00 ~ 22:00

影响：华东地区生产环境存储服务器被误下线。导致语雀数据服务发生严重故障，造成大面积的服务中断。

根因：新的运维升级工具 bug 导致。

修复方案：从备份中恢复数据，新建存储系统

时间线：

14:07 数据存储运维团队收到监控系统报警，定位到原因是**存储在升级中因新的运维工具 bug 导致节点机器下线**；

14:15 联系硬件团队尝试将下线机器重新上线；

15:00 确认因存储系统使用的**机器类别较老，无法直接操作上线**，立即调整恢复方案为从备份系统中恢复存储数据。

15:10 开始新建存储系统，**从备份中开始恢复数据**，由于语雀数据量庞大，此过程历时较长；

19:00 完成数据恢复；同时为保障数据完整性，在完成恢复后，用时 2 个小时进行数据校验；

21:00 存储系统通过完整性校验，开始和语雀团队联调；

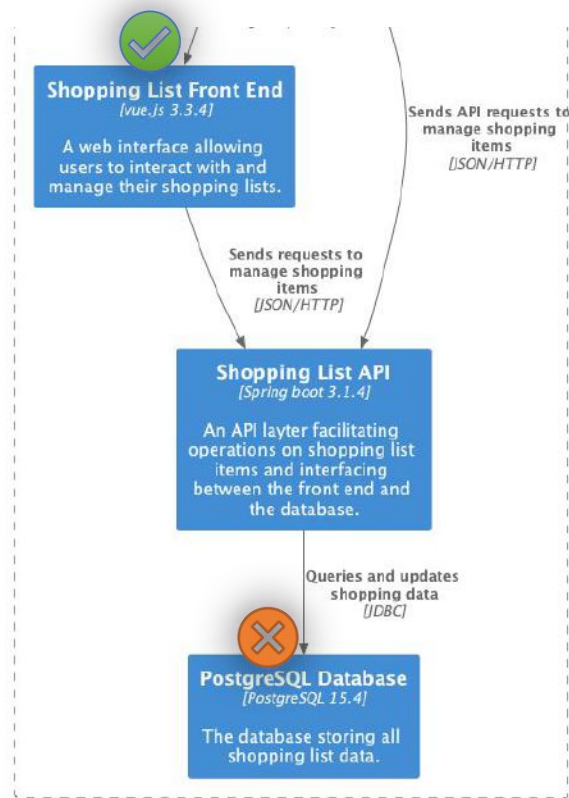
22:00 恢复语雀全部服务，用户所有数据均未丢失





# 可隔离 - 以语雀意外事件为例说明如何实现“意外事件助推避坑”

稳态假说：华东地区生产环境存储服务器被某种原因下线后，在重新上线之前，存储服务器事件点可隔离，即不影响前端应用正常使用，且前端应用仍可向用户报告整个系统现状。





## 可避坑 - 反模式 - 以语雀故障公告为例说明如何撰写“意外事件避坑指南”

- 1 缺少上下文：**只站在意外事件直接负责团队的角度撰写，缺乏“存储系统所使用的机器类别较老，无法直接操作上线”的背景介绍，导致其他团队的读者在阅读避坑指南时，产生误解，难以理解。
- 2 使用术语且不做解释：**避坑指南中使用了“Region”、“多副本容灾”、“两地三中心”等术语，但是没有解释，导致不是做软件的读者在阅读时，难以理解，进而对避坑产生怀疑。
- 3 缺乏意外事件确切的影响数据：**只提到意外事件持续7个多小时，缺乏其他数据，导致难以判断事件所影响的范围等规模程度，也难以判断意外事件何时完全恢复。
- 4 缺乏触发因素的底层细节描述：**只提供了意外事件的高层次描述，难以判断意外事件的最底层根本原因，比如难以了解“在升级中新的运维工具”具体出了什么bug，这难以在将来实现避坑。





## 可避坑 - 反模式 - 以语雀故障公告为例说明如何撰写“意外事件避坑指南” (续)

5 **缺乏意外事件恢复的细节描述**：读者希望了解如何能缓解意外事件的影响的实际措施细节，比如数据恢复、数据校验和联调细节，以便学习和分析。

6 **缺乏可验证的预防措施**：“运维团队加强运维工具的质量保障与测试，杜绝此类运维 bug 再次发生”听起来不错，但是缺乏具体的可执行的措施，更缺乏可验证性，让读者怀疑是否能真正实现。

7 **具有指责意味**：在避坑指南中，提到“运维团队加强运维工具的质量保障与测试，杜绝此类运维 bug 再次发生”，这句话的潜台词是，运维团队没有做好工作，导致意外事件发生。这种指责会促使运维团队产生抵触情绪，会心里反问：“为何语雀不做异地双活？”，难以实现避坑。

8 **分享范围有限**：避坑指南只在内部分享，导致其他团队和用户无法学习和了解，无法重塑信心，也无法提供反馈。（语雀这一点做得较好）

9 **延迟发表**：避坑指南在意外事件发生后，经过了很久才发表，难以实现避坑。（语雀这一点做得较好）





# 可避坑 - 模式 - 以语雀故障公告为例说明如何撰写“意外事件避坑指南”

- 1 提供上下文：**避坑指南中提供意外事件的背景介绍，包括语雀的用户规模、用户分布、用户使用场景、用户使用习惯、用户使用时段等，让读者能够更好地理解意外事件的影响范围。
- 2 解释术语：**避坑指南中解释“Region”、“多副本容灾”、“新的运维工具”等术语，让读者能够更好地理解意外事件。
- 3 提供意外事件确切的影响数据：**避坑指南中提供意外事件的持续时长、影响用户数、影响用户比例、影响用户使用时长、影响用户使用频率等数据，让读者能够更好地理解意外事件的影响范围。
- 4 提供触发因素的底层细节描述：**避坑指南中提供意外事件的底层细节描述，比如运维工具的 bug 如何触发问题的过程，让读者能够更好地理解意外事件的根本原因。
- 5 提供意外事件恢复的细节描述：**避坑指南中提供意外事件的恢复细节描述，比如数据恢复的过程、验证过程、验证结果等，让读者能够更好地理解意外事件的恢复过程。
- 6 提供可验证的预防措施：**避坑指南中提供可验证的预防措施，比如存储服务器离线后可快速上线的预期时长、如何验证运维动作的灰度范围和灰度时长等，让读者能够更好地理解意外事件的预防措施。





## 可避坑 - 模式 - 以语雀故障公告为例说明如何撰写“意外事件避坑指南” (续)

7 **不具有指责意味**：明确系统设计改进已考虑用异地双活应对此类意外事件，没有个人或团队会因该事件而受到指责。重点关注“出了什么问题”，而不是“谁”导致了事件。旨在改善机制和系统，而不是惩罚个人。

8 **分享范围广泛**：将“关于语雀 23 日故障的公告”更名为“语雀23日意外事件相关避坑指南”，有助于让当事人放下难堪而愿意在避坑指南中分享事件细节，让其他团队能够学习和提供反馈。（语雀这一点做得较好）

9 **及时发表**：避坑指南在意外事件发生后，一周之内及时发表，让其他团队能够及时学习，也能够及时提供反馈。（语雀这一点做得较好）

10 **数据驱动结论**：提出的所有结论均基于事实和数据。

11 **提供图表辅助说明**：以图表的形式提供更多有用的信息，以便为了向不熟悉该系统的读者提供背景信息。





# 分布式计算的八大谬误

\*1 网络可靠 -> 依赖可靠;

\*2 延迟为零;

\*3 带宽无限;

\*4 网络安全;

5 拓扑不变;

6 一人管理;

7 传输免费;

8 网络同构。

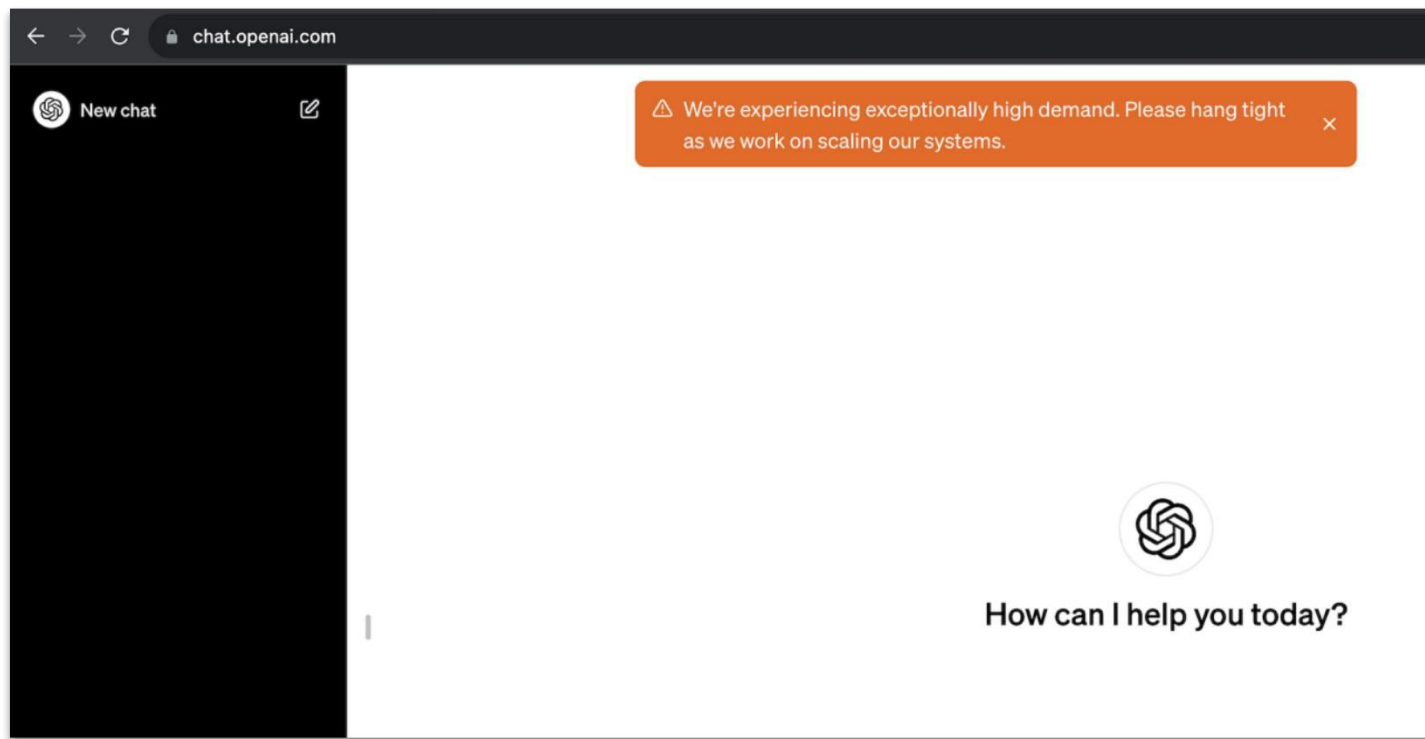
( \*为经常被忽视 )



2023.11.12阿里云意外事件时淘票票用户界面



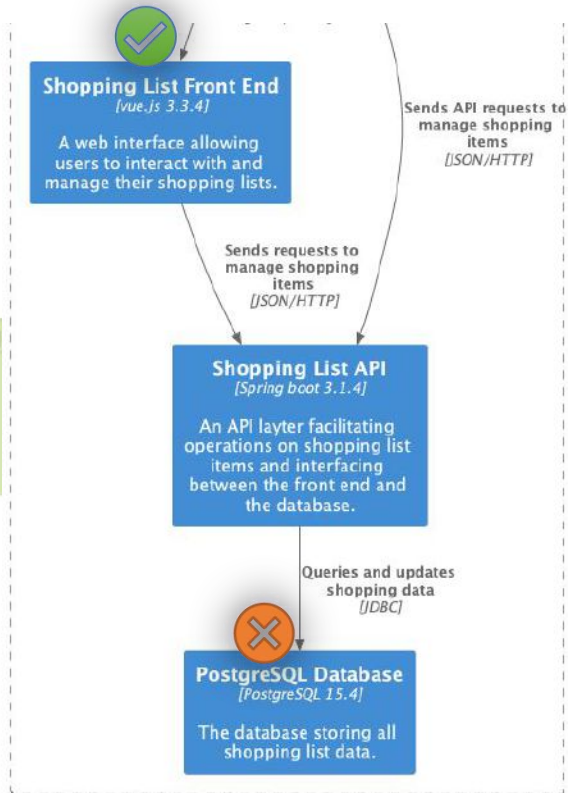
# 榜样：当chatgpt后端无法访问时……





# 可避坑 - 演练 - 针对“依赖可靠”谬误

```
13 async function loadShoppingItems() {
14   try {
15     const response = await axios.get('http://localhost:8081/api/v1/shopping-items')
16     shoppingItems.value = response.data
17     console.log('ShoppingItems', shoppingItems.value)
18   } catch (error) {
19     console.log('An error occurred:', error)
20+   ElMessage({      You, last month • feat: display user-friendly messages on web pag
21+     message: 'Could not connect to the back end app.',
22+     type: 'error',
23+     duration: 5000 // Display the message for 5 seconds
24+   })
25 }
26 }
```

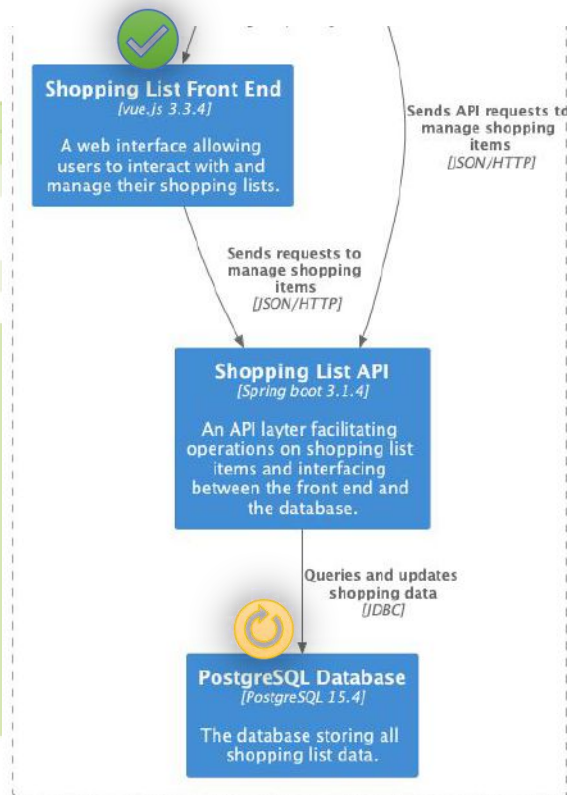






# 可避坑 - 演练 - 针对“延迟为零”谬误

```
13 async function loadShoppingItems() {
14   try {
15+    const response = await axios.get('http://localhost:8081/api/v1/shopping-items', {
16+      timeout: 3000 // Set timeout to 3 seconds
17+    })
18    shoppingItems.value = response.data
19    console.log('ShoppingItems', shoppingItems.value)
20+  } catch (error: any) {
21    console.log('An error occurred:', error)
22+    if (error.code === 'ECONNABORTED') {
23+      ElMessage({
24+        message: 'The response from the back end was delayed for over 3 seconds.',
25+        type: 'error',
26+        duration: 4000 // Display the message for 4 seconds
27+      })
28+    } else {
29+      ElMessage({
30+        message: 'Could not connect to the back end app.',
31+        type: 'error',
32+        duration: 4000 // Display the message for 4 seconds
33+      })
34+    }
35  }
36 }
```





## 总结



# 总结

当前生产事故复盘实践的**三大缺点**

- 1 复盘报告分享难
- 2 演练考试非考验
- 3 缺乏亲身体验感

三大缺点给做软件的人所带来的**三大痛点**:

- 1 难被认可
- 2 心里发虚
- 3 很快遗忘

**“意外事件助推避坑”四大价值点:**

- 1 领导更认可
- 2 更多人学到
- 3 练过心不慌
- 4 印象更深刻

实现”意外事件助推避坑“**原则**:

预防虽重要，难免遇意外。

**修复意外时，系统仍可用。**

不去指责人，实操印象深。

利他且易懂，领导更认可。

业界普遍**缺乏“可隔离”、“可替换”和“可避坑”**这三种有助快速修复的思维



# 源于社区 服务社区



群聊: 吾真本的混沌工程实验室



该二维码 7 天内 (11 月 27 日前) 有效, 重新进入将更新



做软件的吾真本  
北京 西城



扫一扫上面的二维码图案, 加我为朋友。

备注: chaos



中国DevOps社区峰会 2023 · 广州

