

<<编码整洁之道>>

图书基本信息

书名：<<编码整洁之道>>

13位ISBN编号：9787121175633

10位ISBN编号：7121175630

出版时间：2012-8

出版时间：电子工业出版社

作者：罗伯特·C.马丁

页数：244

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## &lt;&lt;编码整洁之道&gt;&gt;

## 前言

1986年1月28日东部时间上午11:39,“挑战者”号航天飞机发射73.124秒后由于右侧固态火箭助推器(SRB)的故障,在48000英尺的高空被撕成碎片。

7名英勇的宇航员,其中包括高中教师克里斯塔·麦考利芙,因此而离开了我们。

麦考利芙的母亲眼看着自己女儿在9英里的高空离她而去时所流露出来的表情直到今天仍然让我无法释怀。

“挑战者”号失事的主因是出了故障的右侧固态火箭助推器中的高热废气泄漏后穿过船体的各段最终飞溅到外部燃料箱上。

接下来,主液氢燃料箱的底部爆炸,燃料被点燃后把箱体推入上面的液氧箱体内。

与此同时,右侧固态火箭助推器脱离其尾部支撑柱并开始围绕其前方支撑柱进行旋转。

而后它的凸起部分刺穿了液氧箱体。

这些异常的力量导致飞行器为了逆气流而旋转 需以远超1.5马赫的速度进行移动,最终空气动力把一切都撕成了碎片。

在右侧固态火箭助推器的环形段之间有两个橡胶的O形环密封圈。

当飞行器的各段被拴在一起时,O形环密封圈将被压缩,并形成严密的密封腔,废气无法穿过。

但是在发射的前一晚,发射台的温度低至17°,比O形环密封圈的最低临界温度低23°,比任何之前的发射低33°。

因此,O形环密封圈变得非常硬以致无法密封住高热气体。

在右侧固态火箭助推器被点燃后,高热气体迅速聚积形成压力脉冲。

而后,助推器的各段向外膨胀并向O形环密封圈释放压力。

O形环密封圈的硬度使它们无法保持密封状态,因此部分高热气体从O形环密封圈的70°弧内漏出和蒸发。

莫顿·塞奥科公司中设计了右侧固态火箭助推器的工程师们其实已经知道O形环密封圈存在这个问题,7年前他们就向莫顿·塞奥科的经理们以及NASA进行过相关报告。

实际上,O形环密封圈在上次发射中已经因为类似的原因被损坏,尽管还不是灾难性的损毁。

结果,最寒冷的发射终于使最严重的伤害成为现实。

虽然工程师们已经为此设计了修复方案,但是修复本身却被一直拖延下来。

工程师们曾经怀疑O形环密封圈会在寒冷中变硬。

他们也知道“挑战者”号的发射温度比以往任何一次都低,并且远低于红线。

也就是说,工程师们已经知道风险非常高。

工程师们根据自己对这些情况的了解而采取行动,他们写下提出警报信号的备忘录。

他们向塞奥科和NASA强烈呼吁不要进行发射。

在一个直到发射前数小时还在持续进行的开了11个小时的会议上,这些工程师展示了最重要的数据。

他们炸开了锅,软磨硬泡,反对发射。

但最终经理们忽略了他们的声音。

当发射时间来临时,一些工程师拒绝观看直播,他们害怕看到发射台上的爆炸。

但是当“挑战者”号优美升空时他们开始放松。

在飞行器解体前,当看到飞行器速度超过1马赫时,他们中有人说:躲过一劫。

尽管有各种抗议和备忘录,以及工程师们的呼吁,经理们仍然相信他们更了解情况。

他们认为工程师们反应过度了,不相信工程师们的数据和推论。

处在巨大的经济和政治压力之下,他们进行了发射,期盼一切都会顺利。

这些经理并不仅是愚蠢,他们是在犯罪。

7位优秀的先生和女士的生命、一代人期待太空旅行的希望,在那个寒冷的早晨皆因那些经理们把自己的恐惧、希望和直觉凌驾于专家意见之上而破灭。

他们做了一个他们并没有权利来做的决定。

他们篡夺了真正了解状况的人——工程师们的权利。

## <<编码整洁之道>>

但是该如何评价工程师们呢？

当然，工程师们做了他们应该做的事情。

他们通知了经理并为自己的责任而进行了斗争。

他们使用了各种合适的渠道和各种权利协议。

他们做了他们所能做的——在特定的系统之下，但最终经理们践踏了这些努力。

因此工程师们或许可以免受责难，而若无其事地走开。

但有时候我很想知道，他们之中是否有人会在夜里因为克里斯塔·麦考利芙母亲的神情而辗转难眠，后悔自己当年没去找过丹·拉瑟。

关于本书 这是一本关于软件专业主义的书。

本书给出了许多务实的建议，并试图回答如下这些问题也： 究竟什么样的人才是软件专家？

一名专家究竟应该如何处事？

专家应该如何处理并应对冲突、紧张的日程以及蛮不讲理的经理？

专家应该在什么时候，用什么样的方式说“不”？

专家会如何面对压力？

但是你会发现书里面隐藏在务实建议背后的是一种斗争并取得突破的态度。

这是一种诚实，珍视荣誉，自尊并自豪的态度。

这是一种愿意接受作为专家和工程师所系重大责任的意愿。

这种责任意味着要把活儿“做好”且“干净利落”。

它意味着有效沟通，据实评估。

它也意味着管理好自己的时间，在风险与回报之间做出审慎的决定。

但是这种责任之中还包含了一些其他的東西——一个可怕的东西。

作为一名工程师，你对你的系统和项目所了解的深度是经理们不可能企及的。

与这种了解相对应，你就也有责任在必要时采取行动。

## <<编码整洁之道>>

### 内容概要

忍受各种不确定性及不间断的压力并能够获取成功的程序员有一个共通特征：他们都深度关注软件创建实践。

他们都把软件看做一种工艺品。

他们都是专家。

在“鲍勃大叔”看来“专业”的程序员不仅应该具备专业的技能，更应该具备专业的态度，这也是本书阐述的核心。

专业的态度包括如何用带着荣誉感、自尊、自豪来面对进行软件开发，如何做好并做得整洁，如何诚实地进行沟通和估算，如何透明并坦诚地面对困难做抉择，如何理解与专业知识相伴的责任。

<<编码整洁之道>>

作者简介

作者:(美)Martin

## 书籍目录

Foreword Preface Acknowledgments About the Author On the Cover Pre-Requisite Introduction Chapter 1 Professionalism Be Careful What You Ask For Taking Responsibility First, Do No Harm Work Ethic Bibliography Chapter 2 Saying No Adversarial Roles High Stakes Being a “ Team Player ” The Cost of Saying Yes Code Impossible Chapter 3 Saying Yes A Language of Commitment Learning How to Say “ Yes ” Conclusion Chapter 4 Coding Preparedness The Flow Zone Writer ’ s Block Debugging Pacing Yourself Being Late Help Bibliography Chapter 5 Test Driven Development The Jury Is In The Three Laws of TDD What TDD Is Not Bibliography Chapter 6 Practicing Some Background on Practicing The Coding Dojo Broadening Your Experience Conclusion Bibliography Chapter 7 Acceptance Testing Communicating Requirements Acceptance Tests Conclusion Chapter 8 Testing Strategies QA Should Find Nothing The Test Automation Pyramid Conclusion Bibliography Chapter 9 Time Management Meetings Focus-Manna Time Boxing and Tomatoes Avoidance Blind Alleys Marshes, Bogs, Swamps, and Other Messes Conclusion Chapter 10 Estimation What Is an Estimate ? PERT Estimating Tasks The Law of Large Numbers Conclusion Bibliography Chapter 11 Pressure Avoiding Pressure Handling Pressure Conclusion Chapter 12 Collaboration Programmers versus People Cerebellums Conclusion Chapter 13 Teams and Projects Does It Blend ? Conclusion Bibliography Chapter 14 Mentoring, Apprenticeship, and Craftsmanship Degrees of Failure Mentoring Apprenticeship Craftsmanship Conclusion Appendix A Tooling Tools Source Code Control IDE/Editor Issue Tracking Continuous Build Unit Testing Tools Component Testing Tools Integration Testing Tools UML/MDA Conclusion Index

## 章节摘录

版权页：插图： A kata usually comes in the form of a simple programming problem to solve, such as writing the function that calculates the prime factors of an integer. The point of doing the kata is not to figure out how to solve the problem; you know how to do that already. The point of the kata is to train your fingers and your brain. I'll do a kata or two every day, often as part of settling in to work. I might do it in Java, or in Ruby, or in Clojure, or in some other language for which I want to maintain my skills. I'll use the kata to sharpen a particular skill, such as keeping my fingers used to hitting shortcut keys, or using certain refactorings. Think of the kata as a 10-minute warm-up exercise in the morning and a 10-minute cool-down in the evening.

**COLLABORATION** The second best way to learn is to collaborate with other people. Professional software developers make a special effort to program together, practice together, design and plan together. By doing so they learn a lot from each other, and they get more done faster with fewer errors. This doesn't mean you have to spend 100% of your time working with others. Alone time is also very important. As much as I like to pair program with others, it makes me crazy if I can't get away by myself from time to time.

**MENTORING** The best way to learn is to teach. Nothing will drive facts and values into your head faster and harder than having to communicate them to people you are responsible for. So the benefit of teaching is strongly in favor of the teacher.

## <<编码整洁之道>>

### 编辑推荐

从《编码整洁之道:专业程序员的行为准则(英文版)》中读者可以学到：一个真的软件专家究竟应该如何处事？

如何处理冲突，安排紧张的日程，以及与不通情理的项目经理打交道？

如何读懂编码的工作流，并了解前人的思路上的断点？

如何面对无情的压力并避免倦怠？

如何客观地对待新的开发范式？

如何管理你的时间并且避免走入死胡同，陷入泥潭？

如何培育可以让程序员个人以及团队茁壮成长的环境？

什么时候要说“不”及如何去说？

什么时候说“是”及“是”究竟意味着什么？



版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>