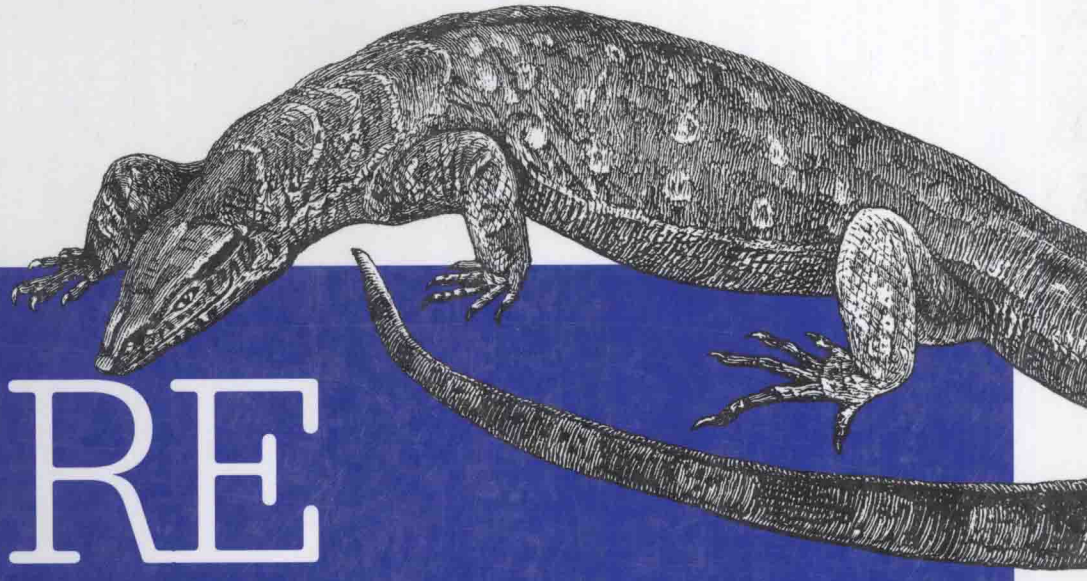
O'REILLY®

Broadview®
www.broadview.com.cn



SRE
Google运维解密

Site Reliability Engineering: How Google Runs Production Systems

[美] Betsy Beyer Chris Jones Jennifer Petoff Niall Richard Murphy 编著
孙宇聪 译

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

O'REILLY®

SRE Google运维解密

Site Reliability Engineering：How Google Runs Production Systems

[美] Betsy Beyer Chris Jones Jennifer Petoff Niall Richard Murphy 编著

孙宇聪 译

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

大型软件系统生命周期的绝大部分都处于“使用”阶段，而非“设计”或“实现”阶段。那么为什么我们却总是认为软件工程应该首要关注设计和实现呢？在本书中，Google SRE的关键成员解释了他们是如何对软件进行生命周期的整体性关注的，以及为什么这样做能够帮助Google成功地构建、部署、监控和运维世界上现存最大的软件系统。通过阅读本书，读者可以学习到Google工程师在提高系统部署规模、改进可靠性和资源利用效率方面的指导思想与具体实践——这些都是可以立即直接应用的宝贵经验。

任何一个想要创建、扩展大规模集成系统的人都应该阅读本书。本书针对如何构建一个可长期维护的系统提供了非常宝贵的实践经验。

©2016 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and Publishing House of Electronics Industry, 2016. Authorized translation of the English edition, 2016 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

本书简体中文版专有出版权由O'Reilly Media, Inc.授予电子工业出版社。未经许可，不得以任何方式复制或抄袭本书的任何部分。专有出版权受法律保护。

版权贸易合同登记号　图字：01-2016-6274

图书在版编目（CIP）数据

SRE：Google运维解密/（美）贝特西·拜尔（Betsy Beyer）等编著；孙宇聪译. —北京：电子工业出版社，2016.10

书名原文：Site Reliability Engineering: How Google Runs Production Systems

ISBN 978-7-121-29726-7

Ⅰ. ①S… Ⅱ. ①贝… ②孙… Ⅲ. ①网站—开发 Ⅳ. ①TP393.092

中国版本图书馆CIP数据核字（2016）第200120号

策划编辑：张春雨

责任编辑：刘　舫

封面设计：Karen Montgomery　张　健

印　　刷：北京京科印刷有限公司

装　　订：北京京科印刷有限公司

出版发行：电子工业出版社

　　北京市海淀区万寿路173信箱　　邮编：100036

开　　本：787×980　1/16　　印张：31　　字数：695千字

版　　次：2016年10月第1版

印　　次：2016年10月第1次印刷

定　　价：108.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819　faq@phei.com.cn。

O'Reilly Media, Inc.介绍

O'Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 Make 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版、在线服务或者面授课程，每一项 O'Reilly 的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar 博客有口皆碑。”

——Wired

“O'Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference 是聚集关键思想领袖的绝对典范。”

——CRN

“一本 O'Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去 Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

赞誉

我们都知道 Google 公司的分布式系统设计和实现在业界遥遥领先，这些分布式系统多年前就已经运行在百万台服务器上，很多公司也都在觊觎这么多服务器是如何运行和管理的。本书揭开了这层神秘的面纱，SRE 就是运行和管理这百万台服务器和众多分布式系统的关键。

多年前，Google 是通过发布技术论文帮助业界解决分布式难题的，如今各种分布式系统百花齐放，如何管理这些系统对传统的运维技术和理念产生了极大的挑战，现在 Google 给我们带来了技术指导和最佳实践。该书汇集了 Google 多年生产环境的管理经验，连编写工作都采用了分布式实现的方法，由各个领域的资深专家联合创作而成。可以把本书看作是一座灯塔，很多公司的集群规模还远达不到 Google 的规模，但是参照本书中的技术指导和最佳实践，不仅可以加速传统运维向 SRE 的进化，更重要的是可以帮助公司高效地运维和管理各种复杂的分布式系统。

——吕宏利，Google Ads SRE

信息技术领域是英文缩写词的高产领域，几乎所有的新概念、新技术和新产品的推出甚至一场市场营销的策划都会伴随着新的英文缩写词的出现。SRE 这个缩写，在公司内部不仅代表了一个全新的运维理念和其伴随的崭新的工程领域、一套完整的系统运维体系和其对应的最佳实践，而且也是我和我的好朋友——本书的译者孙宇聪一起工作了数年的战斗集体。而本书的作者们也都是这个大集体中的师长和伙伴。

系统运维长久以来都依赖实践积累之上的口口相传，经验通常是领域从业者手里掌握的秘诀。本书从实践出发，汇集了众多业内顶尖的系统运维人员的实战心得，理论基础和实操指导并重，系统化地阐述了在新一代信息系统架构（大规模、分布式、高并发、多

试读结束：需要全本请在线购买： www.ertongbook.com

业务、多租户）下系统运维的理念（当前被广泛接受并被大量实践的 DevOps 就起源于此）、思路、最佳实践以及对应的组织架构和人员管理的方方面面，是系统运维领域从业人员不可多得的参考和学习资料。本书是对新时代系统运维领域实践的总结和理论升华。

本书的译者孙宇聪在生活中是一个略显粗犷的大男人，但对于本书的翻译，他充分发挥了自己在这个领域中多年的从业经验和对系统运维的深刻理解，细致入微地做到内容和语言两个方面的精准和优美，这在翻译的技术图书中是非常难得的。

——张矩，锋瑞资本执行董事，前 Google SRE

很高兴受译者孙宇聪邀请为该书写推荐序，这本书是 Google 的 SRE 部门多年实践的总结，孙宇聪本人也在 Google SRE 部门工作多年。SRE 部门在 Google 真正落实了 DevOps。SRE 工程师在 Google 不只是维护各种线上服务的稳定性，还要负责保证各项服务的性能，同时负责管理维护数据中心。美国多家互联网公司都在依照 Google 的方式来组织和运作 SRE 部门，可以说 SRE 被 Google 发扬光大，Google 的 SRE 实践正在成为 DevOps 的标准。

SRE 和传统的 IT 运维有很大区别，SRE 真正实现了 DevOps：首先，SRE 深度参与开发阶段的工作，对应用程序的设计实现方式、依赖库、运行时的资源消耗都有严格的规约；其次，SRE 工程师本身也要做不少编程工作，来实现各种工具用以自动解决问题和故障，换句话说，SRE 强调的是对问题和故障的自动处理，而非人工干预；再者，按照 SRE 的约定，开发人员自行负责程序上线部署更新，毕竟开发人员对自己开发的程序更熟悉，易于处理程序上线过程中遇到的问题。总之，作为 Google 的 DevOps 实践，SRE 非常注重开发和运维职能的结合，极大地加快了业务应用迭代周期，提升了 IT 对业务的支撑能力。

随着 DevOps 在国内的宣传推广，国内的很多企业客户也逐渐接受了 DevOps 的理念，但是在具体落地实践 DevOps 的过程中缺乏实际案例作为参照。本书的推出，方便了国内广大 IT 人员在落地 DevOps 过程中参照 Google 的 SRE 实践。非常感谢孙宇聪把这么好的一本书翻译成中文。

——王璞，数人云创始人

Google 首创了 SRE 这个职业，并将其 SRE 思想体系和方法论贡献出来汇集成此书。中文版的及时出版，使得国内广大运维从业者可以更高效地赏阅并实践。很荣幸此书在 GOPS 全球运维大会首发，高效运维社区将继续作为 Google SRE 国内第一传播平台，推

进其和《互联网应用运维框架及能力模型》（本书译者孙宇聪先生联合撰写）的融合，促进其在中国运维行业的落地生根、蓬勃发展。

——萧田国，高效运维社区发起人，开放运维联盟联合主席

从接触 Google SRE 的概念开始，就感受到它神秘地存在，直到看到英文版的 SRE 书籍，才知道它对传统运维的颠覆性。本书的面世，让国内更多的运维人员接触到 Google 先进的运维理论与实践。个人坚信这种理论和实践的提升与改变，才是运维人的出路，运维的业务价值、行业价值便也随之而来。运维也可以“高大上”地存在!

——王津银，“精益运维”发起人；优维科技创始人；开放运维联盟发起人之一；开放运维联盟应用标准规范组组长、起草人

大型互联网应用的部署规模从几千台到几十万台不一，随着软件系统的复杂度提升也呈现出越来越庞大的趋势，如何通过少数人力管理好庞大复杂的应用环境？如何在环境极度复杂的情况下确保软件的服务质量？如何在确保质量的情况下优化软件迭代速度？很多问题困扰着项目管理者、产品经理、软件工程师、运维人员。本书从 Google 所面临的问题、价值观、解决方案、体系建设、最佳实践等方面理论结合实际，非常具备指导意义，每一个希望提高工作效率、改进工作成果的技术和管理人员都应该认真阅读理解，结合自身工作环境进行实践，找出一条适合自己的持续发展之路。

——莫显峰，Ucloud 联合创始人，CTO

Google 丰富的产品与服务已成为全球多数网民每天生活的一部分，而支撑这许多应用的是其背后庞大的基础设施。为了更有效地保证用户体验，Google 建立了独树一帜的运维体系并称之为 SRE（Site Reliability Engineering）。绝大部分传统 IT 公司会雇佣系统管理员（sysadmin）来运维复杂的计算机系统，但由于大部分工作依靠手工操作，所以随着用户增长，Sysadmin 的团队也必须相应地增长。Google SRE 团队的精华在于研发软件系统，将运维自动化以替代传统模型中的人工操作。这本书详细地描述了 Google SRE 的原则与理念，并列举了实际案例来说明如何灵活运用这些准则。

孙宇聪在 Google 任职八年。他不仅精通基础设施的各个方面，还热衷于钻研平台架构。他致力于为中文读者解析 Google 运维的窍门，于是在繁忙的工作之余，翻译了这本由他的原同事们撰写的书。

由于 Google 的规模很大，许多人可能认为 Google 的做法无法效仿，但书中描述的原则与道理是可以触类旁通的。书中提及许多实用的道理，比如，100% 的可用性是不现实的，需要达到这个目标的成本通常远超于所能获得的价值，所以 Google 会针对每种产品设定一个错误预算（容错率），既能保证用户体验又不影响创新和部署的速度。

我希望读者像我一样，通过阅读这本书，能学习到如何更有效地运维自己的产品与平台。

——Joe Zhu，Zenlayer 创始人

Google SRE 团队通过写作本书为整个运维行业做出了巨大的贡献。通过本书，他们将指导思想、最佳实践和常见的应用架构模式以及团队建设模式共享出来，揭示了 Google 如何能够持续不断地建设、部署世界级的工程项目，同时保持世界一流的可靠性标准。每个感兴趣的人都应该通读本书，切身尝试书里提到的一些想法。

Jez Humble，*Continuous Delivery* 和 *Lean Enterprise* 书籍的共同作者

我还记得 Google 第一次在运维技术论坛上发表的演讲。感觉就像听了一场野生动物专家针对两栖爬行动物的专题介绍。演讲非常有意思，但是由于演讲的内容和观众的日常工作感觉距离太遥远，因此演讲的效果并不好。

随着 IT 行业的不断改变，中小型企业的运维实践逐渐和 Google 接轨。突然之间，Google 多年打磨、积累形成的运维实践变成了最热门的行业焦点。对于一个面临日益严峻的可靠性、可扩展性、可维护性挑战的行业，这本书真是太及时了！

——David N. Blank-Edelman，总监，USENIX 董事会成员，
以及 SREcon 大会的共同创始人

自从我离开 Google 这座充满魔力的城堡，我就一直在等这本书面世，我一直在用书中的思想理念给同事们布道。

——Björn Rabenstein，SoundCloud 生产工程团队负责人，
Prometheus（开源项目）开发者，前 Google SRE（2013）

Google 是 SRE 理念的发明者。本书不光介绍了这个职位的技术细节，还包括了其中的思考过程、团队目标、设计理念以及学到的宝贵课程。如果你想从起源上了解 SRE 一词的意义，应该从本书开始。

——Russ Allbery，Google SRE，安全工程师

本书的作者们和大家分享了 Google SRE 团队的成长经历，包括其中走过的弯路。Google 凭借这些实践经验，将 Google 服务部署到全世界，同时保持世界一流的可靠性。我高度建议任何一个想要创建、扩展大规模集成系统的人阅读本书。这本书针对如何构造一个可长期维护的系统提供了非常宝贵的实践经验。

——Rik Farrow，USENIX 成员

开发一个 Gmail 这样的大型分布式系统已经很难了。如何运营维护这样的一套系统，在保障每天不断更新的同时保障一流的可靠性就更难了。这本书就像一套完备的菜谱，收集了 Google 在实践过程中积累的宝贵经验。希望通过阅读本书，读者能够绕开一些 Google 曾经走过的弯路。

——Urs Holzle，Google 基础架构组资深副总裁

译者序

当我在2016年年初听说本书的英文版即将面世时，第一时间就意识到这将是一本不可多得的经典之作。我作为Google SRE曾经的一员，看到本书中提到的那些熟悉的技术和理念时非常兴奋——现在终于有机会用一种体系化、结构化的方式将这些知识和技术与大家分享了！

Google SRE全球共计约1000人，负责运维Google的大部分家喻户晓、不可或缺的商业应用。同时，SRE还负责运维幕后那些全球首屈一指的计算基础设施，不管是全球百万台级别的服务器集群，还是全球一流的网络架构，背后都有SRE的身影。每个小的传统运维问题在这个平台上似乎都被无限放大了。但是与此同时，Google恰恰又是利用最传统、最朴素的软件工程方法将其一一解决的。

SRE是一群天生的怀疑论者，我们怀疑一切宣传起来“高大上”的技术，以及任何“神奇”的产品——我们只想看具体的设计架构、实现细节，以及真实的监控图表。SRE在保障系统可靠性方面并没有什么万能药，有的只是这种极强的务实态度（pragmatic）。这种务实的态度决定了SRE会认真对待运维问题。在设计评审中，他们会认真推演各种灾难场景。在每周例会时，他们又会讨论如何消除和防范事故发生、优化各种警报策略以及增强自动化功能。在平时工作中，他们则会精心维护团队的各种文档和项目源代码，一点一点地提高服务质量。回头看来，SRE其实是一群崇尚工匠主义的人，我们坚信只要不断地解决根源问题，服务质量就一定会得到提升。而SRE正是用这种“日拱一卒”的方法造就了Google这个世界级的奇迹。

本书的风格亦是如此。书中很多章节用务实的语言记录了Google SRE团队在面临各种困难时的思考过程、所采用的解决方案以及事后总结的经验教训。本书中没有介绍任何“魔法系统”，也没有提供任何“奇技淫巧”，有的只是对问题本质发人深省的深入探讨。从这种意义上讲，本书体系化地覆盖了运维工作的方方面面，是一本运维行业的教科书。我希望通过翻译此书，能将这种体系和理念分享给更多的人。期待与大家更深入地探讨与交流！

回首在 Google 度过的 8 年时光，我想感谢我所有的前同事，感谢他们对我的各种帮助，这段职业经历是我终生难忘的。而且，我还要感谢我的家人，是他们的耐心陪伴和帮助才让我踏踏实实地度过了这 200 多个小时，完成了我人生中最大的一个 Project。

孙宇聪

2016 年 8 月 3 日 傍晚

试读结束：需要全本请在线购买： www.ertongbook.com

前言

如果用一个词语来描述 Google 的历史，那就是不断地“扩大规模”（scaling up）。Google 的成长经历，是计算机行业中数一数二的成功故事，标志着整个社会向 IT 为中心的商业模式的转变。Google 很早就开始实践 IT 与商业模式的结合，也是向社区推广 DevOps 理念的先行者。本书就是由来自公司各个部门，切身参与甚至主导了整个行业转型实践的人写成的。

Google 是在一个系统运维工程师行业转型的阶段发展壮大的。Google 的发展史就像是对传统的系统运维理念发出的革命宣言：我们无法按照传统方式运维 Google 系统，必须要思考一种新的模式，但是同时我们也没有时间等待其他人验证和支持我们的理论。在 *Principles of Network and System Administration*（参见文献 [bur99]）一书的介绍中，我提出了一种观点：系统运维本质上是人与计算机共同参与的一项系统性工程。当时的一些评论者对这种观点表示了强烈的反对：“这个行业还远远没有成熟到可以称为一项工程”。在那时，我甚至对整个运维行业产生了怀疑，认为这个行业在个人英雄主义与神秘色彩中已经永久地迷失了自己，无法前进。但是，这时 Google 诞生了，Google 的高速发展将我的预言变成了现实。我之前的定义变成了一个具体的词语：SRE，站点可靠性工程师。我的几个朋友切身参与了这个新职业的创立，用软件工程理念和自动化工具定义了这个行业。一开始，他们显得很神秘，Google 公司内的体验和整个行业也格格不入，Google 太特殊了！ 随着时间的推移，公司内外交流逐渐增多。这本书的目标就是将 SRE 的一些思考和实践带给整个行业，以促进交流。

在本书中，我们不仅仅展示了 Google 是如何建设维护其富有传奇色彩的大型计算集群的。更重要的是，我们展示了在建设过程中，Google 工程师团队是如何学习、成长、反复修改，最后定义出一套完整的工具和科技体系的过程。IT 行业大多自我封闭，交流过少，很多从业人员都或多或少地受教条主义的限制。如果 Google 工程师团队能克服这个惯性，保持开放的精神，那么我们也能够一起和他们面对 IT 行业内最尖端的挑战。

这本书由一群有共同目标的 Google 工程师写就的短文组成。本书的作者们聚集在同一

个公司旗下，为了同一个目标努力，本身就是一件很特别的事情。在本书的各个章节之间经常能看到软件系统的共同点，以及工作模式上的共通之处。我们经常可以从多个角度分析不同的决策选择，了解他们是如何一起解决公司内部多种利益冲突的。这些文章并不是严谨的学术研究论文，而是这些人的切身经历。虽然本书的作者们有着不同的工作目标、写作风格，以及技术背景，但是他们都尝试着去真诚地描述自己遇到的问题和解决的经历。这和 IT 行业内的普遍文风截然不同，风格迥异。有些作者会宣称："不要做 A，只有做 B 才是正确的。"另一些作者会更谨慎，行文更富有哲学性。这其实恰恰代表了整个 IT 行业内不同个性融合的现状。而我们作为读者，作为观察者，并不了解整个 Google 的历史，也没有参与到具体的决策过程中，无法全面了解当事人所面临的纠缠不清的挑战，只能带着谦逊的态度远远旁观。在阅读本书的过程中，相信读者一定会产生出许多疑问："他们当时为什么没有选择 X？""如果他们选择了 Y，结果会是怎样？""如果多年之后回头再看，这个选择会是正确的吗？"这些问题，恰恰是阅读本书的最大收获，因为我们第一次有机会将自己的经历、选择和本书陈列的决策逻辑相互对应，从中发现不足和缺陷。

这本书的成书过程也堪称奇迹。今天，我们能感受到整个行业都在鼓吹厚颜无耻的"代码拿来主义"（just show me the code）。开源软件社区内部正在形成一种"不要问我问题"的风气，过于强调平等却忽略领域专家的意见。Google 是行业内为数不多的，愿意投入精英力量钻研本质问题的公司，而且这些公司精英很多都有工学博士学位。工具永远只是解决方案中的一个小小组件，用来链接日益庞杂的软件、人和海量的数据。这本书中没有万能药，没有什么东西能解决一切问题，但是这恰恰是本书的宗旨：相比最后的软件结果、架构设计而言，真实的设计过程、作者本身的思考经历更有价值。实现细节永远只是短暂存在的，但是文档化的设计过程却是无价之宝。有机会了解到这些设计的内幕真是太难得了！

这本书，归根到底，记录了 Google 这个公司的成长经历。书中的很多故事都是由相互重叠的故事组成的，这恰恰说明了扩展一个计算机系统，要比简单按照书本上的标准架构放大困难得多。一个公司的成长，意味着整个公司商业模式和工作模式的扩展，而不是简单的资源扩张。仅此一点，这本书就物超所值了。

自省，是一个在 IT 行业内部并不流行的词语。我们不断重复发明各种系统。很多年以来，只有 USENIX LISA 大会论坛以及其他几个专注于操作系统的会议会讨论一些 IT 基础设施的设计和实现。很多年后的今天，IT 行业已经天翻地覆，但是本书仍然弥足珍贵：它详细记录了 Google 迈过分水岭时期的全过程。很显然，这些经历没有办法完全复制，也许只能被模仿，但是却可以启发读者，指引未来。本书作者们表现出了极大的真诚，显示了谦逊的风格，以及极强的凝聚力、领导力。这些文章记录了作者们的希望、担忧、

成功与失败的经历。我向这些作者们和编者们的勇气致敬，正是这种坦率，让我们能够作为旁观者和后来人，从前人的经历中学习到最宝贵的知识。

Mark Burgess
In Search of Certainty 一书作者
Oslo，2016 年 3 月

序言

软件工程有的时候和养孩子类似：虽然生育的过程是痛苦和困难的，但是养育孩子成人的过程才是真正需要花费绝大部分精力的地方。但是，传统软件工程专业花费了很多精力讨论软件的开发过程，而不是其后的维护过程。有统计显示，一个软件系统的40%~90% 的花销其实是花在开发建设完成之后不断维护过程中的。[注1] 行业内流行的一个说法是：一个系统如果已经开发完成，部署在生产环境上，那么它就是“稳定的”，就不需要那么多工程师花费精力去优化、维护。我们认为这个说法是错误的。从这个视角出发，我们认为如果软件工程职业主要专注于设计和构建软件系统，那么应该有另外一种职业专注于整个软件系统的生命周期管理。从其设计一直到部署，历经不断改进，最后顺利退役。这样一种职业必须具备非常广泛的技能，但是和其他职业的专注点都不同。Google 将这个职位称为站点可靠性工程师（SRE，Site Reliability Engineering）。

那么，站点可靠性工程师究竟代表着什么呢？的确，这个词语并不能够特别清晰地描述这个职位的意义。基本上每个 Google SRE 都会被经常问到这个职位到底代表什么意思，以及他们的日常工作究竟是什么。

将这个词语展开来说：首先，也是最重要的一点，SRE 是工程师（engineer）。SRE 使用计算机科学和软件工程手段来设计和研发大型、分布式计算机软件系统。有的时候，SRE 和产品研发团队共同工作，其他时候我们需要开发这些系统的额外组件：例如备份系统和负载均衡系统等。理想情况下，同时推进这些组件在多个项目中复用。还有的时候，我们的任务是想出各种各样的办法用现有组件解决新的问题。

其次，SRE 的关注焦点在于可靠性。Ben Treynor Sloss，Google 负责 7 × 24 运维的副总裁，SRE 名称的发明者，宣称可靠性应该是任何产品设计中最基本的概念：任何一个系统如果没有人能够稳定地使用，就没有存在的意义。因为可靠性[注2] 是如此重要，因此 SRE 专

注 1　在这些预测中数据变动的幅度这么大，本身就说明软件工程不是一个非常注重精确性的职业（具体细节参见文献 [Gla02]）。

注 2　在我们的讨论中，可靠性（reliability）是指某个系统能够在指定环境下，成功持续执行某个功能指定的时间的概率（参见文献 [Oco12] 中的定义）。

注于对其负责的软件系统架构设计、运维流程的不断优化，让这些大型软件系统运行得更可靠，扩展性更好，能更有效地利用资源。但是，SRE 并不是无止境地追求完美：当一个系统已经“足够可靠”的时候，SRE 通常将精力转而投入到研发新的功能和创造新的产品中。[注3]

最后，SRE 的主要工作是运维在分布式集群管理系统上运行的具体业务服务（service）。不论是遍布全球的存储服务，还是亿万用户赖以工作的 E-mail 服务，还是 Google 最初的 Web 搜索服务。SRE 中的“S”最开始指代的就是 *google.com* 的运维服务，因为 SRE 的第一个工作就是维持网站的正常运转。随着时间的推移，SRE 逐渐接管了 Google 内部绝大部分产品系统，包括 Google Cloud Platform 这类开发者平台，也包括内部的一些非网站类的基础设施系统，例如 Bigtable。

虽然我们在这里将 SRE 的职位定义得比较宽泛，但是在这样一个互联网业务高速发展的时代，这个职位的出现毫不奇怪。同样，虽然在应用系统运营维护的过程中有数不清的重要环节需要关注，我们最关注的是“可靠性”这一点也不奇怪。[注4] 在 Web 服务领域里，对服务器端软件的优化和修改是相对可控的，变更管理与生产安全又结合得非常紧密，一种类似于 SRE 的职业早晚会在这个环境下诞生。

虽然 SRE 这个行业是在 Google 内部，从 Web 社区中诞生的，但我们认为这个职业对其他团队和组织也有很多值得借鉴的地方。本书是对阐述 SRE 发展过程的一次尝试：我们既希望将这些宝贵经验共享给其他相关行业，也希望能从其他行业中汲取知识，从而更好地定义各种角色和术语。为了这个目的，本书将通用的理论、设计理念和思想，与实际的应用工具介绍等分开。在某些需要结合 Google 内部信息讨论主题的时候，我们相信读者可以进行类比，将书中的理念与自己的实际环境相结合，以便得出更为有效的结论。

本书中也包含了一些对 Google 内部生产环境的介绍，将 Google 内部环境与外部常见的开源类软件相对应。这样可以让本书的一些设计理念与实践的结合度更强，应用起来更容易。

最后，我们当然希望社区内出现更多、更可靠的软件系统。我们知道，创业企业甚至中型企业经常对如何应用这些理念和技术感到困惑。可靠性就像安全性，越早关注越好。这就意味着一些小型创业公司，在应付日常面临的种种挑战时，也应该抽出一部分精力来面对可靠性这个话题。这与盖房子有些类似，如果一开始将整个地基打好并保持继续修缮，要比盖好房子之后再重新修改设计要容易得多。本书第 4 部分着重介绍了 SRE 团

注 3　我们主要关注的软件系统是大型网站和类似的 Web 服务；这里我们不会讨论大型核电站、民航，以及医疗器械或者其他安全性要求极高的系统。然而，在第 33 章中，我们将自己的方法与这些行业中采用的方法进行了比较。

注 4　在这里，我们故意与行业内流行的词语 DevOps 进行区分。虽然我们认同基础设施即代码的理念，但我们主要关注的是可靠性。同时，我们也更倾向于将运维的需要直接消除，具体细节参见第 7 章。

队如何进行内部培训、如何加强内部沟通等最佳实践，很多都可以直接拿来应用。

对中型企业来说，企业内部可能已经有这样的一组人在做着与 SRE 非常类似的工作。这些人可能并不叫 SRE 这个名字，甚至可能没有受到管理层的重视。在这样的企业中，提高可靠性最好的办法往往就是去认可这些人的工作，并配备足够的激励机制。在牛顿被世界正式认可为物理学家之前，他经常被称作是最后的炼金术师。而这些专注于可靠性的工程师们，正如当年的牛顿一样，是一个新时代的开拓者。

如果一定要为 SRE 寻找一个起源的话，谁才能够被称为世界上第一个 SRE 呢？

我们选择了 Margaret Hamilton，MIT 教授，参与了阿波罗登月计划的软件开发工作。她的工作具有现代 SRE 的一切特性。[注5] 用她自己的话来讲："团队文化就是从一切经历中不断学习，包括来自那些我们最意想不到的地方的经历。"

在 Apollo 7 飞船研发期间的某一天，Margaret 带着她的小女儿 Lauren 一起来到公司。在 Margaret 忙着和组员们在大型计算机上运行飞行模拟测试的时候，她的小女儿偷偷地按下了控制台上的 DSKY 键。整个模拟程序出乎意料地崩溃了，导致整个火箭发射程序意外终止。Margaret 和组员调试后发现，原来 Lauren 意外触发了 P01 这段子程序的执行，导致了整个模拟过程的失败。（该子程序是起飞前调试程序，执行时会删除现存的导航信息，如果在火箭飞行过程中执行这段程序，计算机将无法继续维持火箭航线，后果将是灾难性的。）

凭借着 SRE 的直觉，Margaret 为项目组提交了一个软件改动，申请在飞行程序中增加一项特殊状态检查，以避免飞行员在飞行过程中意外触发 P01 程序的执行。但不幸的是，NASA 管理层认为，这项错误发生的可能性太小，根本不值得为此添加这项修改。于是 Margaret 没有能够成功提交这项软件修改。她只能在火箭飞行手册中添加了一段文字，写道："在飞行过程中，请勿触发 P01 程序。"（当时增加这段文字时，很多 NASA 工程师都认为这很好笑，认为 Margaret 是小题大做，几乎所有人都认为宇航员经过如此长时间的专业训练，是绝对不会犯如此低级的错误的。）

几天后，在 Apollo 8 飞船执行下一项飞行任务时。宇航员 Jim Lovell、William Anders 和 Frank Borman 三人执行一个长达四天的飞行计划途中，Jim Lovell 意外地触发了 P01 程序的执行。更巧的是，当天正好是美国圣诞节，大部分工程师都休假去了。可想而知，当时 NASA 的一片混乱状态。这次不是演习，而是人命关天的危急时刻，如果不能及时解决，三名宇航员将永远无法安全返回了。所幸，当时 Margaret 的飞行手册更新中恰恰提到了这种情形，并且提供了重新上传数据以及恢复执行的有效办法，在有限的时间内解决了问题，使任务可以继续进行。

注 5　在这个故事之外，她同时也参与推广了"软件工程"（Software Engineer）这个词语。

试读结束：需要全本请在线购买： www.ertongbook.com