O'REILLY®

分布式系统应用设计

Designing Distributed Systems

Brendan Burns 著

赵军平 王天青 译

中国电力出版社

分布式系统应用设计

Brendan Burns 著
赵军平 王天青 译

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®
O'Reilly Media, Inc. 授权中国电力出版社出版

中国电力出版社

图书在版编目（CIP）数据

分布式系统应用设计 /（美）布兰登·伯恩斯（Brendan Burns）著；赵军平，王天青译. —北京：中国电力出版社，2019.9

书名原文：Designing Distributed Systems

ISBN 978-7-5198-3485-2

I. ①分… II. ①布… ②赵… ③王… III. ①分布式操作系统－系统设计 IV. ①TP316.4

中国版本图书馆CIP数据核字(2019)第168997号

北京市版权局著作权合同登记 图字：01-2019-0698号

出版发行：中国电力出版社
地　　址：北京市东城区北京站西街19号（邮政编码100005）
网　　址：http://www.cepp.sgcc.com.cn
责任编辑：刘 炽（liuchi1030@163.com）
责任校对：黄蓓 朱丽芳
装帧设计：Randy Comer，张 健
责任印制：杨晓东

印　　刷：北京天宇星印刷厂
版　　次：2019年9月第一版
印　　次：2019年9月北京第一次印刷
开　　本：750毫米×980毫米 16开本
印　　张：10
字　　数：186千字
印　　数：0001—3000册
定　　价：48.00元

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

目录

第III部分 批处理计算模式

前言

谁适合读这本书

如今几乎每个开发人员都堪称为分布式系统的开发人员或使用者（甚至两者皆有）。即使相对简单的移动应用也已经支持了云 API，因此数据可以来自客户手中的任何设备。无论你是开发分布式系统的新兵，还是经验丰富的老手，本书所描述的设计模式和组件都可以帮助你将分布式系统开发从艺术转变为科学。分布式系统的可重用组件和相关开发模式使得开发者可以更专注于应用的核心逻辑。本书的目标是帮助开发人员在构建分布式系统时可以做得更好、更快、更高效。

为何写这本书

从 Web 搜索到云，在我的软件系统开发经历中，曾经负责过许多可扩展、可靠的分布式系统。总的来说，这些系统几乎每一个都是从头开始构建的，而且其他很多分布式应用程序也是如此。虽然秉持着相同的设计理念，有时甚至是几乎相同的逻辑，但如何合理采用合适的设计模式或重用组件，依然是摆在开发者面前极具挑战性的任务。坦白讲，在重新系统实现上我浪费了太多的时间，而结果有时还不如原来那般精致。

最近兴起的容器和容器编排系统则从根本上改变了分布式系统开发的格局。

突然间，我们有了一个更好的对象与接口，来表示核心分布式系统模式及构建可重用的容器化组件。我写这本书主要是为了汇集分布式系统从业者的经验，提供一个可共享的语言和通用标准库，以方便快捷地构建更好的系统。

今天的分布式系统世界

曾几何时，人们编写的程序主要在单机上运行和访问。世界变化如此之快，现在几乎每个大型应用程序都是运行在多台机器上的分布式系统，开放给世界各地的海量用户并发访问。然而设计和开发这些系统通常像个黑盒子，仅由少部分人所掌握。与所有技术类似，分布式系统也在不断地快速演进、规范化和抽象化。本书主要描述了一系列可重复的通用模式，这些模式可以使可靠的分布式系统的开发更加简单和高效。模式和可重用组件的采用使得开发人员无需再重复造轮子，节省的时间则可以专注于构建应用核心逻辑。

本书的主要内容

本书分为以下四部分：

第 1 章，概述

简单介绍分布式系统，并解释为什么模式和可重用组件可以在可靠的分布式系统的快速开发中发挥重要作用。

第 I 部分，单节点模式

第 2~4 章主要讨论在单节点上的设计模式和可重用组件，包括边车、适配器和大使模式。

第 II 部分，服务模式

第 8 章和第 9 章主要针对长期运行的服务（如 Web 应用程序）介绍多节点分布式模式，包括复制、伸缩和主节点选举等。

第 III 部分，批处理计算模式

第 10~12 章介绍用于大规模批处理数据处理的分布式系统模式，包括工作队列、基于事件的处理和协调的工作流程。

如果你是一位经验丰富的分布式系统工程师，可以跳过前面几章，这些章节主要是介绍本书的期望，如何应用这些模式，以及为什么分布式系统模式概念如此重要。

几乎每个人都能在单节点模式中找到一些很有帮助的实用工具，它们是本书最为通用和最可重用的模式。

根据你的学习目标和兴趣点，可以选择专注于大规模的大数据模式，或长期运行服务器模式（或两者兼有）。这两部分在很大程度上彼此独立，可以按任意顺序阅读。

如果你已经有了丰富的分布式系统经验，可能会发现前面某些章节（例如，第II部分中关于命名、发现和负载平衡部分）已经非常熟悉，可以选择性阅读，但推荐快速浏览下其中的图片或者框图。

排版约定

本书使用以下排版约定：

斜体（Italic）

表示一个新术语、URL、电子邮件、文件名或者文件扩展名。

等宽字体（`Constant width`）

表示程序列表及段落内部用于引用程序元素，如变量或函数名称、数据库、数据类型、环境变量、函数语句和关键字等。

加粗等宽字体（**`Constant width bold`**）

表示由用户所输入的命令或其他文本。

斜体等宽字体（Constant width italic）

表示应由用户所提供的实际值或根据上下文所确定的值来替换的文本。

表示提示、建议或一般性补充说明。

表示警告或提醒。

在线资源

本书描述了一般适用的分布式系统模式，但希望读者熟悉容器和容器编排系统。如果你没有很多这方面的知识储备，建议使用以下资源快速学习：

- *https://docker.io*
- *https://kubernetes.io*
- *https://dcos.io*

使用代码示例

相关补充材料（包括代码示例，练习等）可以从 *https://github.com/brendandburns/designing-distributed-systems* 下载。

本书是为了帮助你完成你的工作。通常对于本书提供的示例代码，你可以在自己程序和文档中使用它。除非复制了大部分代码，否则无需与我们联系以获得许可。使用本书中几个代码块不需要许可，出售或分发 O'Reilly 书籍中的示例光盘则需要获得许可，通过引用本书和示例代码来回答一些问题则不需要许可，将本书中的大量示例代码集成到产品文档中则需要获得许可。

我们很希望但并不强制要求你在引用本书内容时加上引用说明。引用说明一般包括书名、作者、出版社和 ISBN。例如：“Designing Distributed Systems by Brendan Burns (O'Reilly). Copyright 2018 Brendan Burns, 978-1-491-98364-5”。

如果你认为对代码示例的使用可能会超出合理的使用范围或上述许可，请通过邮件 *permissions@oreilly.com* 与我们联系。

O'Reilly Safari

Safari（之前称为 Safari Books Online）是一个基于会员的培训和参考平台，适用企业、政府、教育工作者和个人。

会员可以访问来自 250 多家出版商的数千本书籍、培训视频、学习路径、互动教程和策划播放列表，包括 O'Reilly Media、Harvard Business Review、Prentice Hall Professional、Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Adobe、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett 和 Course Technology 等。

想了解更多信息，请访问 *http://oreilly.com/safari*。

如何联系我们

如有任何关于本书的反馈和意见，请联系出版商：

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街2号成铭大厦C座807室（100035）
奥莱利技术咨询（北京）有限公司

我们提供了一个本书的 Web 页面，其中列出了勘误表、示例和相关内容信息。欢迎访问 *http://bit.ly/designing-distributed-systems*。

如有相关评论或技术问题，请发送电子邮件至 *bookquestions@oreilly.com*。

有关我们的书籍、课程、会议和新闻的更多信息，请访问我们的网站 *http://www.oreilly.com*。

我们的 Facebook：*http://facebook.com/oreilly*。

我们的 Twitter：*http://twitter.com/oreillymedia*。

我们的 YouTube：*http://www.youtube.com/oreillymedia*。

致谢

感谢我的妻子 Robin 和我的孩子们，她们让我感到快乐和高效，并为此奉献了很多，也非常感谢那些花时间帮助我学习这些知识的所有人，同时还要感谢我的父母。

第 1 章

概述

在当今 24h 在线的应用和 API 的世界中，对可用性和可靠性有很高要求，几十年前我们仅对全球范围内的少数关键服务才有这样的要求。同样对于那些快速增长甚至是病毒式增长的业务，则意味着面对海量用户的请求，被构建来支撑业务的应用能够即时扩展并响应。这些约束和要求意味着几乎所有被构建的应用程序（无论是针对消费者的移动应用还是后台的支付应用），都必须是一个分布式系统。

但构建分布式系统具有挑战性。很多情况下这些系统都是一次性定制的解决方案。在现代面向对象编程语言的发展面前，分布式系统的发展与软件开发领域具有惊人的相似性。幸运的是，与面向对象语言发展一样，技术的进步已经大大减少了构建分布式系统的挑战。这源于容器和容器编排技术的日益普及。与面向对象编程中的对象概念一样，这些容器化构建模块是开发可重用组件和模式的基础，这些组件和模式大大简化并使构建可靠的分布式系统变得容易。下面我们将简要介绍这些技术目前的发展情况。

系统开发简介

计算机最初是为特定目的而建造的，例如计算火炮表或潮汐，破译密码，或其他精确、复杂，但只需机械计算的数学计算应用。随着时间的推移，这些专用计算机演变成为通用可编程的计算机。同时它们从一次只能运行一个程

序发展到通过分时操作系统可以在一台机器上运行多个程序，但是这些机器仍然彼此互不相连。

渐渐地，计算机开始彼此联网，这时客户端－服务器架构诞生了，它使得低配的计算机能够使用位于另一个房间或建筑物中的大型机的强大功能。虽然这种客户端－服务器编程方式比为单个机器编写程序要复杂一些，但理解它仍然相当简单：客户端发送要求，服务器处理这些请求。

21 世纪初期，由数以千计的、相对低成本的商用计算机组成的互联网和大规模数据中心的崛起使得分布式系统得到了广泛发展。不同于客户端－服务器架构，分布式应用程序系统由在不同计算机上运行的多个不同应用程序或跨不同计算机运行的多个程序副本组成，它们通过网络进行通信，以实现诸如 Web 搜索或零售平台之类的系统。

由于分布式的特性，当架构设计合理时，分布式系统本质上会非常可靠。当正确构建时，它们可以为构建这些系统的软件工程师团队带来更加可扩展的组织模型。然而这些优点是有代价的，分布式系统正确的设计、构建和调试可能会非常复杂。构建可靠的分布式系统所需的软件开发技能明显高于构建移动或 Web 前端等单机应用程序所需的技能。无论如何，对可靠的分布式系统的需求只会继续增长。因此业界迫切需要构建分布式系统的工具、模式和实践。

幸运的是，方便构建分布式系统的技术也在不断发展。容器、镜像和容器编排器，这些构建可靠分布式系统的基础和构建块，近年来也变得流行。以容器和容器编排为基础，我们可以建立一套模式和可重用组件。这些模式和组件是可以用于更可靠、更有效地构建分布式系统的基础。

软件开发中的模式简介

这已经不是第一次在软件行业发生这样的转变。为了更好地理解模式、实践和可重用的组件是如何重塑以前的系统开发的，审视类似转换发生的历史是很有帮助的。

算法编程的形式化

相对于以前人们几十年的编程实践，Donald Knuth 在 1962 年出版的《The Art of Computer Programming》（Addison-Wesley Professional）标志着计算机科学发展的重要篇章。特别地，这本书包含的算法并不是为某一特定的计算机系统设计的，而是为了让读者了解算法本身，然后将这些算法运用于所使用的机器的特定体系结构或读者正在解决的特定问题。这种形式化非常重要，因为它为用户提供了用于构建程序的共享工具包，同时也因为它表明存在一种程序员应该学习并随后可以应用于各种不同上下文的通用概念。独立于任意待解决的具体问题，算法本身就值得了解。

面向对象编程的模式

Knuth 的书成为计算机编程思想中的一个重要里程碑，而算法则成为计算机编程发展的重要组成部分。随着程序复杂性的增加，编写单体程序的人数从个位数增长到百位数，最终增加到数以千计，很明显过程式编程语言和算法已经不足以完成现如今的编程任务。计算机编程领域中的这些变化引起了面向对象编程语言的发展，其在计算机程序的开发中将数据、可重用性和可扩展性提升到和算法一样重要的程度。

为了应对计算机编程的这些变化，编程的模式和实践也发生了变化。20 世纪 90 年代早期到中期，关于面向对象编程模式书籍的数量激增，其中最著名的是由 Erich Gamma 等四人编写的《Design Patterns: Elements of Reusable Object-Oriented Programming》（Addison-Wesley Professional）。设计模式为编程任务提供了通用的语言和框架，它描述了一系列基于接口的模式，可以在各种环境中重用。由于面向对象编程的发展和特定的接口，这些模式也可以实现为通用的可重用库。这些库可以由社区的开发人员编写一次并重复使用，从而节省时间并提高可靠性。

开源软件的兴起

虽然开发人员共享源代码的概念几乎从计算机出现就已经存在，并且自 20 世

纪 80 年代中期以来，正式的自由软件组织就已经存在，但在 20 世纪 90 年代末和 21 世纪初期，开发和分发开源软件的数量才急剧增加。虽然开源不只与分布式系统模式的发展相关，但从某种意义上说，通过开源社区的发展，我们越来越意识到软件开发特别是分布式系统的开发是开源社区努力的结果。值得注意的是，构成本书所述模式基础的所有容器技术都是作为开源软件开发和发布的。从社区角度来看，描述和改进分布式软件开发实践的模式尤为重要。

分布式系统的模式是什么呢？ 有很多文档可以说明如何安装特定的分布式系统（例如 NoSQL 数据库），同样也包括特定系统集合的说明（如 MEAN 堆栈[译注 1]）。但是当谈到模式时，我们指的是构成分布式系统的一般蓝图，而没有强制要求选择任何特定的技术或应用程序。模式的目的是提供一般建议或结构来指导设计。希望这些模式能够指导大家思考，并且适用常见的各种应用程序和环境。

模式，实践和组件的价值

在你花费宝贵时间阅读我认为会改善你的开发实践的一系列模式，教你新技能，从而可能改变你的生活（我有理由相信）之前，我们有理由问："为什么？什么是可以改变我们设计和构建软件方式的设计模式和实践呢？"在本节中我将列出原因，我认为这是一个非常重要的话题，并希望说服你坚持读完本书的其余部分。

站在巨人的肩膀上

作为一个起点，分布式系统模式的价值在于给我们一个站在巨人肩膀上的机会。我们解决的问题或构建的系统很少是真正独特的（也就是之前没有出现过的）。最终我们整合在一起的各个部分及软件所支持的整体商业模式可能

译注 1：MEAN 是用于开发动态网站和网络应用的一套基于 JavaScript 的软件栈，由 MongoDB、Express.js、Angular 和 Node.js 组成。MEAN 的名字是由四个组件首字母组成的。