

互联网大公司一线程序员分享网站架构建设之道，
珍贵实战经验，讲解通俗精练！

Broadview[®]
www.broadview.com.cn

- 系统** 从头梳理网站架构重点难点，零基础上手架构设计
- 全面** 涵盖多种类型网站应用，适合不同类型的读者
- 权威** 凝聚作者5年一线经验，互联网大厂实践方案尽收眼底
- 深入** 剖析概念内涵外延，讲解方法追本溯源
- 案例** 列举10个大型案例，100多个小案例，提升更迅速



深入浅出大型网站 架构设计

李力非 / 著



中国工信出版集团



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn



深入浅出大型网站 架构设计

李力非 / 著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

为了帮助有一定编程基础的读者快速了解如何以职业标准开发一个网站，本书从架构设计的角度出发，涵盖了以高性能、高可用、高并发等多个业内标准为目标网站设计和建设手段，并在每个方面追本溯源，从理论方法到生产实践，在力求简明易懂、适用于尽可能多的场合的前提下深入实践中，为读者提供实用操作指南。

同时，本书对所提出的概念进行了简明扼要的解释，并对介绍的手段和方案不仅解释了如何做，也解释了来源和选择理由，使得读者在理解内容并应用的同时，也能理解这些手段和方案背后的思路，将来可脱离书本，设计出属于自己的创新方案，真正做到了“授人以鱼不如授人以渔”。

本书兼具理论和实践，既可作为网站架构设计的参考书和工具书，又可作为实践指南。

未经许可，不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有，侵权必究。

图书在版编目（CIP）数据

深入浅出大型网站架构设计 / 李力非著. —北京：电子工业出版社，2020.6

ISBN 978-7-121-35397-0

I. ①深… II. ①李… III. ①网站建设—软件开发—架构 IV. ①TP393.092.2

中国版本图书馆 CIP 数据核字（2020）第 076683 号

责任编辑：张月萍 特约编辑：田学清

印 刷：天津千鹤文化传播有限公司

装 订：天津千鹤文化传播有限公司

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：787×1092 1/16 印张：16

字数：314 千字

版 次：2020 年 6 月第 1 版

印 次：2020 年 6 月第 1 次印刷

定 价：89.00 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 zltz@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。

前言

为什么要写这本书

很多刚从学校毕业的计算机专业的学生，或者通过自学掌握编程技能的非计算机专业的人，往往会发现软件工程师在工作中所做内容与学校中所学的知识有不小的差异，并且这种差异随着项目规模的增大而增大。一些拥有不错编程基础的从业人员往往也要在从业数年以后，才能逐渐通过积累工作经验来缩小、弥补这种差异。

造成这种差异的主要原因在于，在学校中学习的编程技能侧重于计算机科学的原理及基本的应用，而在工作中，对于一个工程项目软件，为了使其达到商用、大规模使用的条件，软件工程师会采用许多学校中不会重点教学甚至完全不会接触的方式来确保其开发、维护上的高效率和健壮性。本书在网站开发方面，通过总结笔者从业中遇到过的众多案例和项目，精练出一系列职业经验和操作规程，帮助感兴趣的初学者对职业实践有所了解，而编程能力原本就扎实的程序员更可以通过本书获得职场上的即战力。

本书有何特色

1. 涵盖了大型网站建设从理论到实践的方方面面

对于每个相关的业内实践问题，本书都会涵盖其常见解决方案和最佳推荐选择，并从原理出发分析和解释每个方案，使读者充分理解积累了大量从业人员经验教训的实现方案。

2. 每个主题都包含了大量实例说明

本书为暂时没有机会参与大流量网站建设的读者提供了大量生产实例，使读者能更直观地理解大型网站相比普通网站在生产环境中面临的问题及其解决方案。

3. 对多个实践类主题都附有可直接使用的代码示例

本书对所有可以独立尝试实践的主题都附有实践说明和代码示例，绝大多数代码示例都可以经过简单配置直接使用，非常适合看了相关章节跃跃欲试的读者。

4. 主题鲜明，易于上手

本书各章主题之间尽量保证分割清晰，大多数章节相互独立，且每个主题深入浅出，读者即使不读完全书，只对中间某个主题感兴趣，也可以随手拿起书深入其中一章，而不受上下文的牵制影响。

5. 信息涉及范围广，拓宽视野，与时俱进

本书所提及的设计方案、技术手段和实践标准均与当前业内实践看齐，并在必要部分对业内流行的技术进行了或深或广的介绍，帮助读者轻松地以本书为出发点，找到合适的拓展资料和值得进一步深挖的技术。

本书内容及知识体系

第 1~2 章主要介绍了网站架构的设计目标和原则，包括高性能、高可用、伸缩性和扩展性，并以此为纲展开全书。除此之外，还介绍了软件工程标准的网站架构设计流程。

第 3~7 章主要介绍了网站数据层的几种优化手段，从按需选择数据库到分库分表、读写分离、缓存和动静分离，逐步深入，从数据库介绍到分库分表和读写分离，从缓存介绍到动静分离，先理论再实践，完成数据层的优化改造。

第 8~10 章主要从负载均衡、异步和非阻塞、队列三个角度介绍了如何管理和优化一个网站的整体架构，使其达到高并发，瞬时承担更大流量。

第 11~15 章主要介绍了大型商用网站最重要的性质之一——高可用，以及如何做到高可用。分别从高可用的指导原则、异地多活、服务降级、限流和下游错误处理的角度，解释了在单机服务不可靠的情况下，如何通过架构设计使系统整体变得可靠又稳定。

第 16~17 章主要介绍了大型商用网站在上线服务之前，所需要采取的合理合规的必要手段，以及如何从一个成熟的业务拥有者的角度，尽可能降低新服务上线的风险。

适合阅读本书的读者

- 有一定技术水平但工程资历尚浅的人员。
- 有一定工程资历但没有大流量网站开发和维护经验的人员。
- 希望对网站开发的软件工程有所了解的人员。
- 广大 Web 开发程序员。
- 希望提高大型项目设计水平的人员。
- 软件开发项目经理。
- 需要一本案头必备查询手册的人员。

阅读本书的建议

- 没有网站开发经验的人员，建议配合一本网站开发实战类参考书来阅读。
- 有网站开发经验的人员，可以随意从一章开始阅读，相信都会有所受益。
- 对于有代码示例的章节，可以随时按照其中的说明进行实战；没有代码示例但有操作流程的章节，需要一定生产环境支撑其方法的实践，请读者结合实际工作环境进行学习。

目 录

第 1 章 网站架构概述.....	1
1.1 网站的基本组件.....	1
1.2 网站业务规模增长带来的问题.....	2
1.3 大型网站架构设计的目标和原则.....	4
1.3.1 高性能.....	4
1.3.2 高可用.....	5
1.3.3 伸缩性.....	6
1.3.4 扩展性.....	7
第 2 章 大型网站架构设计的流程.....	9
2.1 需求分析.....	9
2.1.1 需求驱动的重要性.....	9
2.1.2 如何根据需求制定系统目标.....	10
2.2 方案设计.....	11
2.2.1 与架构设计原则相结合.....	11
2.2.2 设计多套备选方案.....	12
2.3 方案评估.....	13
第 3 章 数据库的选择.....	15
3.1 关系数据库.....	15
3.1.1 什么是关系数据库.....	16
3.1.2 关系数据库的优势和应用场景.....	17
3.2 非关系数据库.....	18
3.2.1 什么是非关系数据库.....	18
3.2.2 非关系数据库的优势和应用场景.....	19
3.3 常见的关系数据库产品.....	20
3.3.1 MySQL.....	20
3.3.2 MS SQL Server.....	21

3.3.3 Oracle.....	22
3.4 常见的非关系数据库产品	22
3.4.1 MongoDB	23
3.4.2 DynamoDB	23
3.5 云数据库.....	23
第4章 数据库优化：分库分表.....	25
4.1 什么是分库分表	25
4.1.1 分库	25
4.1.2 分表	26
4.2 为什么要进行分库分表	27
4.2.1 吞吐量	27
4.2.2 索引	27
4.2.3 备份	28
4.2.4 其他风险	28
4.3 实现分库分表.....	28
4.3.1 垂直分库分表.....	29
4.3.2 水平分库分表.....	30
4.4 分库分表带来的问题	32
4.4.1 全局唯一 ID	32
4.4.2 关系数据库的部分操作.....	33
4.4.3 事务支持	33
第5章 数据库优化：读写分离.....	34
5.1 什么是读写分离	34
5.2 为什么要使用读写分离	35
5.2.1 何时需要使用读写分离.....	35
5.2.2 读写分离的好处.....	36
5.3 实现读写分离	37
5.3.1 中间件实现.....	37
5.3.2 应用层实现.....	38
5.4 读写分离带来的问题	39
5.4.1 副本的实时性.....	39

5.4.2	副本实时性的解决方案.....	39
5.4.3	成本问题.....	40
第6章	缓存.....	41
6.1	什么是缓存.....	41
6.2	缓存策略.....	42
6.2.1	LFU 缓存策略.....	42
6.2.2	LRU 缓存策略.....	43
6.2.3	缓存策略的优劣.....	43
6.3	缓存命中率.....	44
6.4	缓存的类型.....	44
6.4.1	客户端缓存.....	44
6.4.2	CDN 缓存.....	45
6.4.3	应用缓存.....	45
6.4.4	基于分布式集群的缓存.....	45
6.5	分布式缓存.....	46
6.5.1	分布式缓存的应用场景.....	46
6.5.2	分布式缓存的架构设计.....	47
6.6	缓存的问题.....	47
6.6.1	缓存过热.....	47
6.6.2	缓存穿透.....	48
6.6.3	缓存雪崩.....	48
6.7	常见的缓存系统.....	49
6.7.1	MemCached.....	49
6.7.2	Redis.....	49
第7章	动静分离.....	50
7.1	动静分离.....	50
7.1.1	动态数据和静态数据.....	50
7.1.2	动静分离的概念.....	52
7.1.3	动静分离的作用.....	53
7.2	拆分动态数据和静态数据.....	55
7.2.1	识别动态数据和静态数据.....	55

7.2.2	改造数据	56
7.2.3	改造数据要注意的问题.....	60
7.3	动静分离的架构改造	62
7.3.1	动静分离的缓存架构.....	62
7.3.2	浏览器缓存.....	63
7.3.3	CDN 缓存	64
7.3.4	Web 服务器缓存	65
7.3.5	分布式缓存.....	65
7.3.6	页面组装	66
第 8 章	负载均衡.....	67
8.1	什么是负载均衡.....	67
8.1.1	负载均衡的概念.....	67
8.1.2	负载均衡的类型.....	69
8.1.3	有负载均衡的网站架构.....	69
8.1.4	反向代理	70
8.2	DNS 负载均衡.....	72
8.2.1	DNS	73
8.2.2	A 记录.....	73
8.2.3	CName	73
8.2.4	配置 DNS 负载均衡.....	74
8.2.5	DNS 负载均衡的优缺点.....	75
8.3	硬件负载均衡.....	76
8.4	软件负载均衡: LVS.....	77
8.4.1	LVS 架构	77
8.4.2	LVS 的负载均衡方式	78
8.4.3	LVS 的负载均衡策略	80
8.4.4	LVS 的调整升级	81
8.4.5	LVS 的优缺点	81
8.5	软件负载均衡: Nginx.....	82
8.5.1	Nginx 架构.....	82
8.5.2	Nginx 的工作原理.....	83
8.5.3	Nginx 的负载均衡策略.....	84

8.5.4	Nginx 的错误重试.....	85
8.5.5	Nginx 的调整升级.....	85
8.5.6	Nginx 的主要特点.....	86
8.5.7	Nginx 配置实战.....	86
8.6	负载均衡的实践流程.....	89
8.6.1	回顾流量基本概念.....	90
8.6.2	实践流程.....	90
第 9 章	异步和非阻塞.....	93
9.1	异步及其相关概念.....	93
9.1.1	同步和异步.....	94
9.1.2	阻塞和非阻塞.....	94
9.1.3	多线程.....	96
9.2	异步和非阻塞的作用.....	97
9.2.1	异步和非阻塞的应用场景.....	97
9.2.2	异步和非阻塞的架构.....	102
9.2.3	异步的优势.....	103
9.3	实战: 以 Java 为例.....	105
9.3.1	Runnable.....	105
9.3.2	Callable.....	106
9.3.3	Future.....	106
9.3.4	Executor 和 ExecutorService.....	108
9.3.5	改造同步且阻塞的 Java 代码.....	108
9.4	异步和非阻塞带来的问题.....	112
9.4.1	API 定义.....	113
9.4.2	线程池的扩容.....	113
第 10 章	队列.....	116
10.1	队列及其相关概念.....	116
10.1.1	队列.....	116
10.1.2	生产/消费、发布/订阅与主题.....	117
10.2	队列与网站的整合.....	119
10.2.1	发布者.....	119

10.2.2	订阅者	120
10.2.3	订阅者：推送模式.....	120
10.2.4	订阅者：拉取/轮询模式.....	122
10.3	队列的应用.....	123
10.3.1	流量控制.....	123
10.3.2	服务解耦.....	126
10.4	队列存在的问题与解决方案.....	128
10.4.1	消息积压.....	128
10.4.2	消息的可靠传递.....	130
10.4.3	消息重复.....	133
10.5	常见的队列产品和系统.....	134
10.5.1	RabbitMQ	134
10.5.2	ActiveMQ	135
10.5.3	RocketMQ.....	135
10.5.4	Kafka.....	136
10.5.5	AWS SQS 和 SNS	136
第 11 章	高可用.....	137
11.1	CAP 原理.....	137
11.1.1	什么是 CAP 原理.....	137
11.1.2	CAP 原理与网站服务.....	138
11.2	服务可用性的标准.....	141
11.3	冗余和隔离.....	142
11.3.1	扩容中的冗余.....	142
11.3.2	广义的冗余.....	142
11.3.3	隔离.....	142
第 12 章	异地多活	144
12.1	异地多活的基本概念.....	144
12.1.1	基本概念.....	144
12.1.2	作用	145
12.1.3	应用场景.....	145
12.1.4	异地多活和负载均衡.....	147

12.2 异地多活的类型.....	147
12.2.1 同城异地多活.....	147
12.2.2 跨城市异地多活.....	148
12.2.3 跨地区异地多活.....	149
12.3 如何进行异地多活改造.....	149
12.3.1 业务分类.....	149
12.3.2 数据分类.....	150
12.3.3 数据同步.....	151
12.3.4 异地多活的数据同步提升方案.....	153
第 13 章 服务降级	156
13.1 服务降级的基本概念.....	156
13.1.1 什么是服务降级.....	156
13.1.2 单点故障.....	158
13.2 微服务与服务拆分.....	160
13.2.1 什么是微服务.....	160
13.2.2 流量模式.....	161
13.2.3 如何拆分服务.....	162
13.3 系统分级.....	165
13.3.1 分析系统流程图.....	165
13.3.2 一级系统.....	166
第 14 章 限流	168
14.1 限流的基本概念.....	168
14.1.1 什么是限流.....	168
14.1.2 为什么需要限流.....	169
14.1.3 限流的几种标准.....	171
14.1.4 限流的几种思路.....	172
14.2 限流算法.....	176
14.2.1 令牌桶算法与漏桶算法.....	176
14.2.2 时间窗口算法.....	179
14.2.3 队列法	182

14.3 服务限流需要考虑的问题.....	183
14.3.1 性能和准确性.....	183
14.3.2 如何进一步提升.....	184
14.4 实战：使用 Nginx 限流.....	186
第 15 章 下游错误处理.....	191
15.1 超时机制.....	191
15.2 错误分类.....	192
15.2.1 如何分类错误.....	192
15.2.2 早期失败.....	194
15.2.3 默认值的作用.....	194
15.3 错误重试.....	195
15.3.1 错误重试的条件.....	196
15.3.2 错误重试带来的问题.....	196
第 16 章 测试.....	198
16.1 测试的类型.....	198
16.1.1 一般功能测试.....	198
16.1.2 黑盒和白盒测试.....	200
16.1.3 不同程度的功能测试.....	202
16.1.4 非功能的测试.....	204
16.2 测试用例的设计.....	206
16.2.1 模拟实际环境.....	206
16.2.2 包含错误情况.....	207
16.2.3 保证用例多样性.....	209
16.2.4 验证系统间的连接性.....	212
16.3 功能测试详解.....	213
16.3.1 单元测试.....	213
16.3.2 集成测试.....	217
16.3.3 端到端测试.....	219

第 17 章 上线准备	222
17.1 发布流程.....	222
17.1.1 规范化流程.....	222
17.1.2 结合测试的流程.....	224
17.1.3 自动化的流程.....	225
17.2 监控.....	226
17.2.1 生产环境度量.....	226
17.2.2 监控与警报.....	231
17.3 压力测试.....	232
17.3.1 压力测试的目的.....	233
17.3.2 如何进行压力测试.....	233
17.4 灰度发布.....	237
17.4.1 什么是灰度发布.....	237
17.4.2 灰度发布的条件.....	239
17.5 维护人员.....	241
17.5.1 应急预案.....	241
17.5.2 人工监控.....	242

第 1 章

网站架构概述

互联网的发展今非昔比，现在稍微成规模的公司、组织、机构甚至个人，都可以建立自己的网站。现在的网站，也早已不是只有一些普通资讯类页面和一些论坛回帖的网站，它们大部分拥有复杂的业务逻辑、优秀的互动界面，其功能非常强大。

在这种情况下，搭建一个网站，目标就不只是搭建一个能运行起来或能展示页面的网站，而是需要进行细致且专业的规划和分析，搭建出一个灵活且功能全面的网站。

本章主要涉及的知识点如下。

- 网站的基本组件有哪些。
- 当网站业务规模变大时，会带来哪些问题。
- 当设计大型网站架构时，应该考虑哪些方面，以什么为目标。

1.1 网站的基本组件

互联网诞生之初，网站的架构都极其简单，所有功能都放在一起，而现在的网站，往往会根据每部分所负责的任务的不同，分成多个组件。究其原因，过去的互联网并不普及，网站能做的事情也不多，一般是对一些个人或组织的信息进行展示，没有承担高流量的要求，而且逻辑简单，没有细分的必要。

现在的网站在公司或组织中扮演着极其重要的角色，是公司或组织业务的线上版本，甚至是公司与用户的主要接口。这些网站的使用频率高，逻辑也极其复杂，所以有必要进行进一步的细分，以达到提升性能、灵活伸缩、更容易变更业务逻辑等的目的。

现在，网站的基本架构如图 1.1 所示。

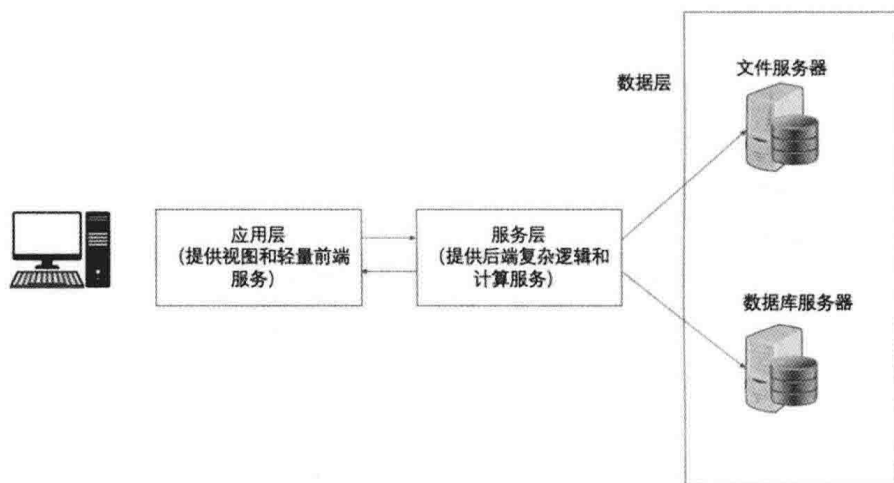


图 1.1 网站的基本架构

1.2 网站业务规模增长带来的问题

本书讨论的重点是大型网站的架构设计。大型网站和普通网站的区别是什么？

所谓量变引起质变，对于一个系统来说，同一件事情，做一次造成的影响和做一百万次造成的影响是完全不同的。

每年春节，国内的公路交通、铁路交通系统都会面临“春运”的压力。而“春运”和平时运输的不同之处，主要就是人流量是平时的成千上万倍。

以铁路为例，这个差距对于买票、检票、安检、列车服务等每一环乃至列车本身造成的压力都是完全不同的，而在出错时，导致的问题则更加复杂。

例如，一列列车出现问题无法出发。在平时人流量小时，车站可以和铁路系统方面进行协调，在下一列列车中找到空位安排受影响的乘客，或者将一部分乘客转给公路系统，或者办理退换票等。但是在“春运”时，所有列车都在运行，退换票业务也极为繁忙，而且车站的人流量非常大，一旦出现整整一车旅客滞留，造成的后果是难以估量的。

这时候如果车站和铁路系统没有对如此大流量的情况有针对性地设计处理方案，不知将有多少旅客不能及时回家与家人团聚。更不用说，这类意外还会对其他车次带来多么大的影响，以及随之而来的安全问题等。

笔者在上文对“春运”进行了大段的阐述和分析，是因为公共交通和运输系统与网站，从存在目的到遇到的问题乃至相应的解决方案，在本质上都极为相似，甚至可以说在很多方面，网站就是运输系统在虚拟世界的一个投射。因此，如果读者能够理解透彻“春运”这样的现实问题，便可以更容易地理解本书的内容，甚至在某些问题