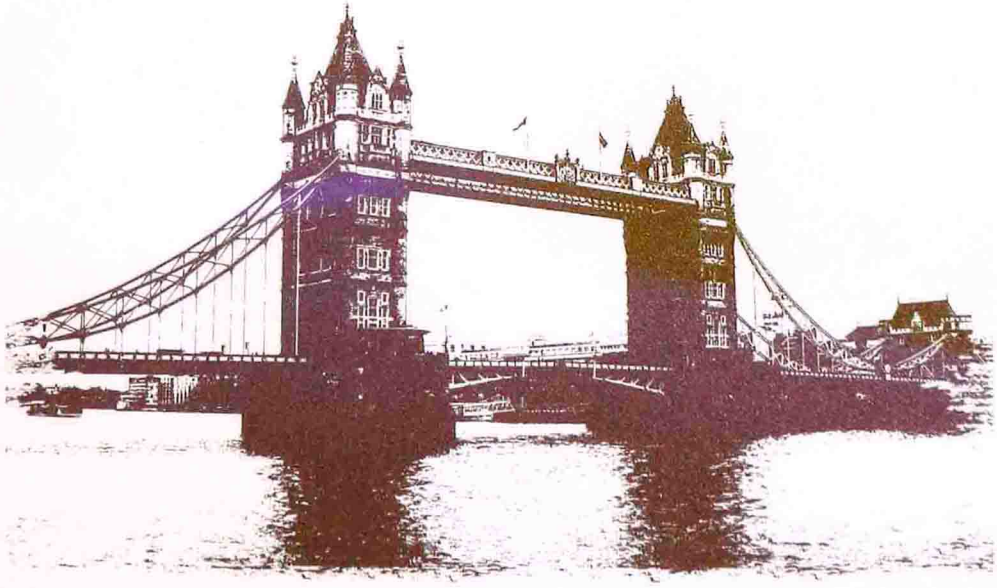
Broadview®
www.broadview.com.cn

大型分布式网站架构
设计与实践

陈康贤 著



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

大型分布式网站架构
设计与实践

陈康贤 著

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

内 容 简 介

本书主要介绍了大型分布式网站架构所涉及的一些技术细节，包括 SOA 架构的实现、互联网安全架构、构建分布式网站所依赖的基础设施、系统稳定性保障和海量数据分析等内容；深入地讲述了大型分布式网站架构设计的核心原理，并通过一些架构设计的典型案例，帮助读者了解大型分布式网站设计的一些常见场景及遇到的问题。

作者结合自己在阿里巴巴及淘宝网的实际工作经历展开论述。本书既可供初学者学习，帮助读者了解大型分布式网站的架构，以及解决问题的思路和方法，也可供业界同行参考，给日常工作带来启发。

图书在版编目（CIP）数据

大型分布式网站架构设计与实践 / 陈康贤著. —北京：电子工业出版社，2014.9

ISBN 978-7-121-23885-7

Ⅰ. ①大… Ⅱ. ①陈… Ⅲ. ①网站－建设 Ⅳ. ①TP393.092

中国版本图书馆 CIP 数据核字（2014）第 169308 号

策划编辑：董　英
责任编辑：陈晓猛
印　　刷：北京中新伟业印刷有限公司
装　　订：三河市皇庄路通装订厂
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：28.75　　字数：640 千字
版　　次：2014 年 9 月第 1 版
印　　次：2014 年 9 月第 1 次印刷
定　　价：79.00 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。

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。

服务热线：（010）88258888。

序

大型分布式网站技术全貌

2008 年，淘宝网随着访问量/数据量的巨增，以及开发人员的增长，原有的架构体系已经无法支撑，于是在那一年淘宝网将系统改造为了一个大型分布式的网站。作者目前就职于阿里集团，清晰地看到了目前淘宝这个大型分布式网站的架构体系，这个架构体系其实是非常多方面的技术的融合，要掌握好最重要的首先是看清全貌，但这也是最难的。本书向大家展示了一个大型分布式网站需要的技术的全貌。

翻看各大型网站的架构演变过程，会发现其中有一个显著的共同点是在某个阶段网站的架构体系改造为服务化的体系，也就是常说的 SOA，SOA 系统之间以服务的方式来进行交互，这样就保证了交互的标准性，对于一个庞大的多人开发的网站而言这至关重要，所以在实现 SOA 时重要的第一点是实现基本的服务方式的请求/响应，除了这点外，对于访问量巨大的网站而言，主要都是采用可水平伸缩的集群方式来支撑巨大的访问量，这会涉及在服务交互时需要做负载均衡的处理，较为简单的一种方式是采用硬件负载均衡设备，这种方式一方面会增加不少成本，另一方面会导致单点的巨大风险，因此目前各大网站多数采用软件负载的方式来实现服务的交互，如何去实现 SOA 是大型分布式网站的必备基础技能。

基于服务化主要是为了解决网站多元化、开发人员增加及访问量增加带来的水平伸缩问题，但数据量增长会带来更多复杂的问题，大型网站都是严重依赖缓存来提升性能的，数据量增长带来的效应就是单机会无法缓存所有的数据，需要引入分布式的缓存；数据量增长对于持久型的存储而言就更为复杂，通常会需要采用分库分表、引入 NoSQL 等方式来解决，对于带来的

模糊查询等需求就更加复杂了，而现在的大型网站多数都有很多用户产生的数据，这也就导致了随着访问量的增长，多数情况下用户产生的数据量也会暴涨，因此数据量增长带来的这些问题也是必须学会如何去解决的。

近几年以来网站的安全形势越来越严峻，这里有一个关键的原因是多数开发在安全方面了解的知识比较少，导致开发的系统在安全上会非常欠缺考虑，但其实对于一个网站而言，安全是基本，尤其是电子商务类网站，一旦出现安全问题，很容易丧失难得建立起来的信任，因此在开发一个大型网站的时候安全的意识非常重要。

网站的稳定性是衡量一个网站的重要指标，对于一个大型网站而言，网站一两个小时不可用会引起严重的公众事件，如何去保障一个庞大的网站的稳定性，涉及不少技术知识。要保障网站的稳定性，首先最重要的是监控，要清楚地知道网站目前的运行状况、有问题的点的状况等，没有监控的网站就像是一辆没有油表的车；监控主要是帮助发现问题，在出现问题后最重要的不是去找到 bug 并修复，而是如何有效快速地恢复，例如最典型的有效手段是优雅降级（也就是 James Hamilton 那篇著名的《On Designing and Deploying Internet-Scale Services》中的 Gracefully Degrade）；在快速恢复了后，则需要定位出造成问题的根本原因，这需要很多经验、扎实的基本功和对各类排查工具的掌握。

对于一个大型网站而言，最宝贵的部分通常是积累下来的数据，怎样用上这些数据来提升帮助业务，除了对商业玩法的掌握外，技术上的难度也非常高，大数据技术也是近几年的热门话题，离线计算、实时流式计算等，都是现在的火热话题，涉及的技术点也非常多，对于一个大型网站的技术掌控者而言，这也是需要了解的知识点。

对于一个大型网站的架构师而言，最重要的是掌控一个网站的技术发展过程，很好地去控制每个阶段需要做什么，并确保每个阶段需要的技术布局是完善的，避免有空白点，相信本书涵盖的方方面面的知识点会给读者提供有效的帮助。

林昊（http://hellojava.info）
阿里巴巴集团资深技术专家
写于阿里巴巴西溪园区
2014 年 8 月

前　言

在大型网站架构的演变过程中，集中式的架构设计出于对系统的可扩展性、可维护性及成本等多方面因素的考虑，逐渐被放弃，转而采用分布式的架构设计。分布式架构的核心思想是采用大量廉价的 PC Server，构建一个低成本、高可用、高可扩展、高吞吐的集群系统，以支撑海量的用户访问和数据存储，理论上具备无限的扩展能力。分布式系统的设计，是一门复杂的学问，它涉及通信协议、远程调用，服务治理，系统安全、存储、搜索、监控、稳定性保障、性能优化、数据分析、数据挖掘等各个领域，对任何一个领域的深入挖掘，都能够编写一本篇幅不亚于本书的专门书籍。本书结合作者在阿里巴巴及淘宝网的实际工作经历，重点介绍大型分布式系统的架构设计，同时，为避免过度专注于理论而使得内容显得空洞，作者穿插介绍了很多实践的案例，尽量让每一个关键的技术点都落到实处，相信能够帮助读者更好地理解本书的内容。

内容大纲

全书共 5 章，章与章之间几乎是相互独立的，没有必然的前后依赖关系，因此，读者可以从任何一个感兴趣的专题开始阅读，但是，每一章的各个小节之间的内容是相互关联的，因此，最好按照原文的先后顺序阅读。

第 1 章主要介绍企业内部 SOA（Service Oriented Architecture，即面向服务的体系结构）架构的实现，包括 HTTP 协议的工作原理，基于 TCP 协议和基于 HTTP 协议的 RPC 实现，如何实现服务的路由和负载均衡，HTTP 服务网关的架构。

第 2 章主要介绍如何保障互联网通信的安全性，包括一些常见攻击手段的介绍；常见的安

全算法，如数字摘要、对称加密、非对称加密、数字签名、数字证书的原理和使用；常用通信认证方式，包括摘要认证、签名认证，以及基于 HTTPS 协议的安全通信；另外还介绍了通过 OAuth 协议的授权过程。

第 3 章介绍一些分布式系统所依赖的基础设施，包括分布式缓存，持久化存储。持久化存储又涵盖了传统的关系型数据库 MySQL，以及近年来开始流行 NOSQL 数据库如 HBase、Redis，消息系统及垂直化搜索引擎等。

第 4 章介绍如何保障系统运行的稳定性，包括在线日志分析、集群监控、流量控制、性能优化，以及常用的 Java 应用故障排查工具和典型案例。

第 5 章介绍如何对海量数据进行分析，包括数据的采集、离线数据分析、流式数据分析、不同数据源间的数据同步和数据报表等。

本书并不假设读者在 Java 领域有很深的技术水平，但是，结合作者本人的工作经验和使用习惯，书中的大部分案例代码均采用 Java 来编写，并且运行在 Linux 环境之上，因此，读者最好对 Java 环境下的编程有一定的了解，并且熟悉 Linux 环境下的基本操作，以便能够更加顺利地阅读本书。

致谢

首先，要感谢我的家人，特别是我的妻子，在我占用大量周末、休假的时间进行写作的时候，能够给予极大的宽容、支持和理解，并对我悉心照顾且承担起了全部的家务，让我能够全身心地投入到写作之中，而无须操心一些家庭琐事，没有你的支持和鼓励，这本书是无法完成的。

同时，要感谢阿里巴巴及淘宝网，给我提供了合适的环境和平台，使自己的技能能够得以施展，并且，身处在一群业界的技术大牛中间，也得到了很多学习和成长的机会。另外，还要感谢我的主管飞悦对于写作开明的态度，以及一直以来的鼓励与支持，并在日常的工作中给予我的很多帮助。

最后，还要感谢博文视点的编辑们，本书能够这么快出版，离不开他们的敬业精神和一丝不苟的工作态度。

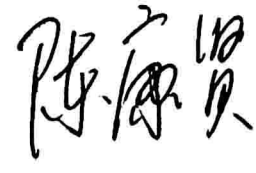
感悟

一年多以前，在接到编辑约稿即将开始动笔之前，自己曾信心满满地认为，应该能够比较顺利地完成这本书，因为写的内容自己都比较熟悉，而且平时工作当中也有一些笔记积累，不是从零开始的。但当真正开始写了以后才知道，理解领悟和用文字表达出来完全是两个层面的

事情，日常工作中一些很普遍很常见的设计思路，可能是由一次次失败和挫折得到的经验教训演变而来。很多时候我们只知道 how，而忽略了 what 和 why，要解释清楚 what、why、how，甚至是 why not，并没有想象中的那么容易。当然，通过写作的过程，自己也将这些知识点从头到尾梳理了一遍，对这些知识的认识和理解也更加深入和全面。每次重新回过头来审阅书稿时，都会觉得某些知识点讲述得还不够透彻，需要进行补充，抑或是感觉对某些知识点的叙述不够清晰和有条理，还能够有更好的表述方式。但是，书不能一直写下去，在本书完稿之时，自己并没有想象中那样的兴奋或者放松，写作时的那种“战战兢兢，如履薄冰”的感觉，依然萦绕在心头，每一次落笔，都担心会不会因为自己的疏忽或者理解上的偏差，从而误导读者。由于时间的因素和写作水平的限制，书中难免会有错误和疏漏之处，恳请读者批评和指正。如有任何问题或者是建议，也可以通过如下方式与作者联系：

博客：chenkangxian.iteye.com

微博：http://weibo.com/u/2322720070



2014 年 5 月于杭州

目　录

本章主要介绍和解决如下问题：

- 分布式缓存 memcache 的使用及分布式策略，包括 Hash 算法的选择。
- 常见的分布式系统存储解决方案，包括 MySQL 的分布式扩展、HBase 的 API 及使用场景、Redis 的使用等。
- 如何使用分布式消息系统 ActiveMQ 来降低系统之间的耦合度，以及进行应用间的通信。
- 垂直化的搜索引擎在分布式系统中的使用，包括搜索引擎的基本原理、Lucene 详细的使用介绍，以及基于 Lucene 的开源搜索引擎工具 Solr 的使用。

本章主要介绍和解决如下问题：

- 常见的 Web 攻击手段和防御方法，如 XSS、CRSF、SQL 注入等。
- 常见的一些安全算法，如数字摘要、对称加密、非对称加密、数字签名、数字证书等。

- 如何采用摘要认证方式防止信息篡改、通过数字签名验证通信双方的合法性，以及通过HTTPS协议保障通信过程中数据不被第三方监听和截获。
- 在开放平台体系下，OAuth协议如何保障ISV对数据的访问是经过授权的合法行为。

本章主要介绍和解决如下问题：

- 常用的在线日志分析命令的使用和日志分析脚本的编写，如cat、grep、wc、less等命令的使用，以及awk、shell脚本的编写。

- 如何进行集群的监控，包括监控指标的定义、心跳检测、容量评估等。
- 如何保障高并发系统的稳定运行，如采用流量控制、依赖管理、服务分级、开关等策略，以及介绍如何设计高并发系统。
- 如何优化应用的性能，包括前端优化、Java 程序优化、数据库查询优化等。
- 如何进行 Java 应用故障的在线排查，包括一系列排查工具的使用，以及一些实际案例的介绍等。

本章主要介绍和解决如下问题：

- 分布式系统中日志收集系统的架构。
- 如何通过 Storm 进行实时的流式数据分析。
- 如何通过 Hadoop 进行离线数据分析，通过 Hive 建立数据仓库。
- 如何将关系型数据库中存储的数据导入 HDFS，以及从 HDFS 中将数据导入关系型数据库。
- 如何将分析好的数据通过图形展示给用户。

第 1 章
面向服务的体系架构（SOA）

伴随着互联网的快速发展和演进，不断变化的商业环境所带来的五花八门、无穷无尽的业务需求，使得原有的单一应用架构越来越复杂，越来越难以支撑业务体系的发展。因此，系统拆分便成了不可避免的事情，由此演变为垂直应用架构体系。

垂直应用架构解决了单一应用架构所面临的扩容问题，流量能够分散到各个子系统当中，且系统的体积可控，一定程度上降低了开发人员之间协同和维护的成本，提升了开发效率。

但是，当垂直应用越来越多，达到一定规模时，应用之间相互交互、相互调用便不可避免。否则，不同系统之间存在着重叠的业务，容易形成信息孤岛，重复造轮子。此时，相对核心的业务将会被抽取出来，作为单独的系统对外提供服务，达成业务之间相互复用，系统也因此演变为分布式应用架构体系[1]，如图 1-1 所示。

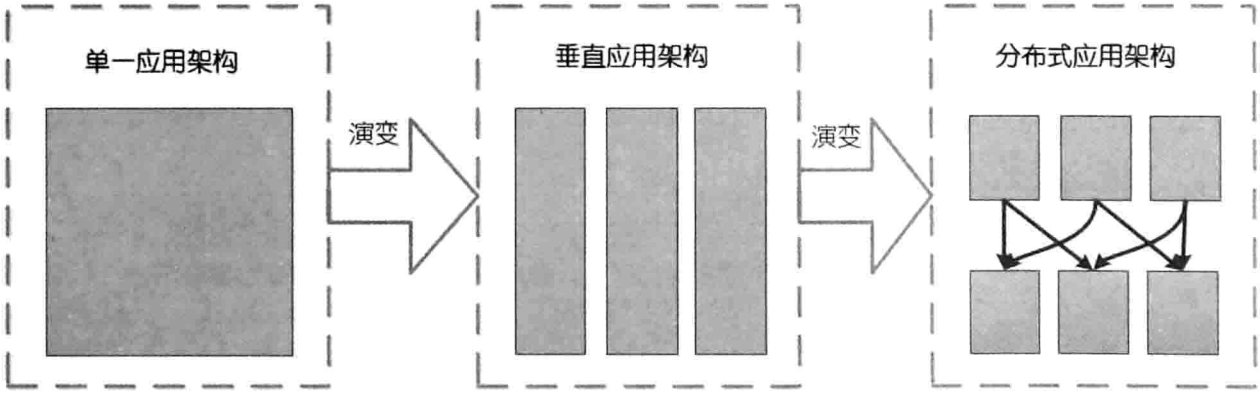


图 1-1 分布式应用架构的演变

分布式应用架构所面临的首要问题，便是如何实现应用之间的远程调用（RPC）。基于 HTTP 协议的系统间的 RPC，具有使用灵活、实现便捷（多种开源的 Web 服务器支持）、开放（国际标准）且天生支持异构平台之间的调用等多个优点，得到了广泛的使用。与之相对应的是基于 TCP 协议的实现版本，它效率更高，但实现起来更加复杂，且由于协议和标准的不同，难以进行跨平台和企业间的便捷通信。当服务越来越多时，使得原本基于 F5、LVS 等负载均衡策略、服务地址管理和配置变得相当复杂和烦琐，单点的压力也变得越来越大。服务的动态注册和路由、更加高效的负载均衡的实现，成为了亟待解决的问题。

本章主要介绍和解决以下问题，这些也是全书的基础：

- HTTP 协议的工作方式与 HTTP 网络协议栈的结构。
- 如何实现基于 HTTP 协议和 TCP 协议的 RPC 调用，它们之间有何差别，分别适应何种场景。
- 如何实现服务的动态注册和路由，以及软负载均衡的实现。

1 分布式系统的演进及服务治理可以参照 http://code.alibabatech.com/wiki/display/dubbo/User+Guide-zh。

1.1 基于 TCP 协议的 RPC

1.1.1 RPC 名词解释

RPC 的全称是 Remote Process Call，即远程过程调用，它应用广泛，实现方式也很多，拥有 RMI[2]、WebService[3]等诸多成熟的方案，在业界得到了广泛的使用。

单台服务器的处理能力受硬件成本的限制，不可能无限制地提升。RPC 将原来的本地调用转变为调用远端的服务器上的方法，给系统的处理能力和吞吐量带来了近似于无限制提升的可能，这是系统发展到一定阶段必然性的变革，也是实现分布式计算的基础。

如图 1-2 所示，RPC 的实现包括客户端和服务端，即服务的调用方与服务的提供方。服务调用方发送 RPC 请求到服务提供方，服务提供方根据调用方提供的参数执行请求方法，将执行结果返回给调用方，一次 RPC 调用完成。关于调用方发起请求及服务提供方执行完请求的方法后返回结果的过程，和所涉及的调用参数及响应结果的序列化和反序列化操作，下节将详细介绍。

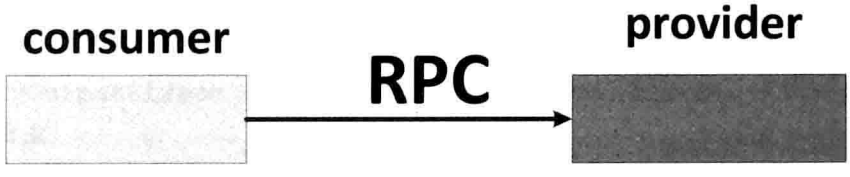


图 1-2 RPC 调用示意图

随着业务的发展，服务调用者的规模发展到一定阶段，对服务提供方的压力也日益增加，因此，服务需要进行扩容。而随着服务提供者的增加与业务的发展，不同的服务之间还需要进行分组，以隔离不同的业务，避免相互影响，在这种情况下，服务的路由和负载均衡则成为必须要考虑的问题，如图 1-3 所示。

服务消费者通过获取服务提供者的分组信息和地址信息进行路由，如果服务提供者为一个集群而非单台机器，则需要根据相应的负载均衡策略，选取其中一台进行调用，有关服务的路由和负载均衡，后续章节会详细介绍，此处不再赘述。

2 RMI，http://en.wikipedia.org/wiki/Java_remote_method_invocation。

3 WebService，http://en.wikipedia.org/wiki/Webservice。

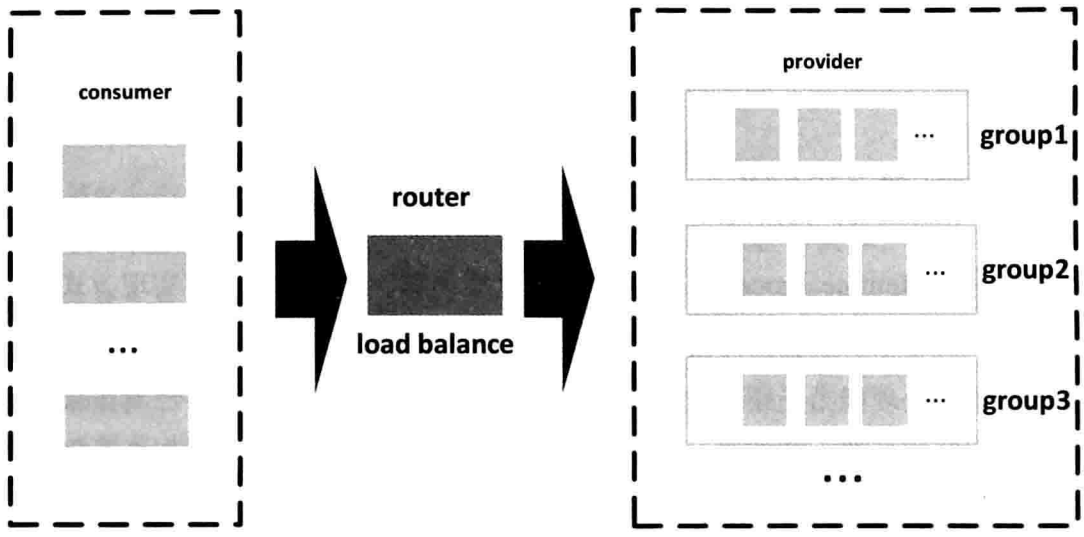


图 1-3 服务的分组路由与负载均衡架构

1.1.2 对象的序列化

无论是何种类型的数据，最终都需要转换成二进制流在网络上进行传输，那么在面向对象程序设计中，如何将一个定义好的对象传输到远端呢？数据的发送方需要将对象转换成为二进制流，才能在网络上进行传输，而数据的接收方则需要把二进制流再恢复为对象。

- 将对象转换为二进制流的过程称为**对象的序列化**。
- 将二进制流恢复为对象的过程称为**对象的反序列化**。

对象的序列化与反序列化有多种成熟的解决方案，较为常用的有 Google 的 Protocal Buffers、Java 本身内置的序列化方式、Hessian，以及后面要介绍的 JSON 和 XML 等，它们各有各的使用场景和优/缺点。图 1-4 所展示的是使用各种序列化方案将一个对象序列化为一个网络可传输的字节数组，然后再反序列化为一个对象的时间性能对比。

Google 的 Protocol Buffers 真正开源的时间并不长，但是其性能优异，在短时间内引起了广泛的关注。其优势是性能十分优异，支持跨平台，但使用其编程代码侵入性较强，需要编写 proto 文件，无法直接使用 Java 等面向对象编程语言的对象。相对于 Protocol Buffers，Hessian 的效率稍低，但是其对各种编程语言有着良好的支持，且性能稳定，比 Java 本身内置的序列化方式的效率要高很多。Java 内置的序列化方式不需要引入第三方包，使用简单，在对效率要求不是很敏感的场景下，也未尝不是一个好的选择。而后面章节要介绍的 XML 和 JSON 格式，在互联网领域，尤其是现在流行的移动互联网领域，得益于其跨平台的特性，得到了极为广泛的应用。

本节重点介绍 Java 内置的序列化方式和基于 Java 的 Hessian 序列化方式，并用代码演示具体实施方法。