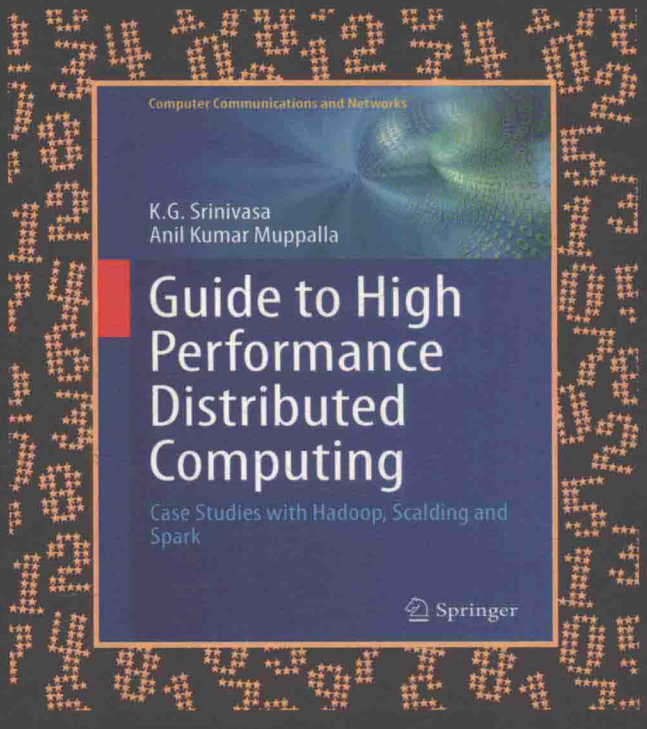
HZ BOOKS 华章IT
Springer
数据科学与工程技术丛书

# 高性能分布式计算系统开发与实现

## 基于Hadoop、Scalding和Spark

[印度] K. G. 斯里尼瓦沙（K .G. Srinivasa）
阿尼尔·库马尔·穆帕拉（Anil Kumar Muppalla） 著

高辉 李东升 王宏志 译




GUIDE TO HIGH PERFORMANCE DISTRIBUTED COMPUTING

Case Studies with Hadoop, Scalding and Spark

机械工业出版社
China Machine Press

数据科学与工程技术丛书

# GUIDE TO HIGH PERFORMANCE DISTRIBUTED COMPUTING

Case Studies with Hadoop, Scalding and Spark

# 高性能分布式计算系统开发与实现

## 基于Hadoop、Scalding和Spark

[印度] K. G. 斯里尼瓦沙（K .G. Srinivasa）
阿尼尔·库马尔·穆帕拉（Anil Kumar Muppalla）

高辉 李东升 王宏志 译



机械工业出版社
China Machine Press

**图书在版编目（CIP）数据**

高性能分布式计算系统开发与实现：基于 Hadoop、Scalding 和 Spark/（印）K. G. 斯里尼瓦沙（K. G. Srinivasa），（印）阿尼尔·库马尔·穆帕拉（Anil Kumar Muppalla）著；高辉，李东升，王宏志译 . —北京：机械工业出版社，2018.7
（数据科学与工程技术丛书）
书名原文：Guide to High Performance Distributed Computing: Case Studies with Hadoop, Scalding and Spark

ISBN 978-7-111-60153-1

I. 高… II. ① K… ② 阿… ③ 高… ④ 李… ⑤ 王… III. 分布式数据处理 IV. TP274

中国版本图书馆 CIP 数据核字（2018）第 138279 号

**本书版权登记号：图字 01-2017-2015**



本书介绍了如何使用开源工具和技术开发与实现大规模分布式处理系统，涵盖构建高性能分布式计算系统的方法和最佳实践。第一部分（第 1～4 章）介绍了高性能分布式计算编程的基础知识，包括分布式系统、Hadoop 入门、Spark 入门、Scalding 入门等；第二部分（第 5～8 章）给出了使用 Hadoop、Spark、Scalding 的案例研究，涉及数据聚类、数据分类、回归分析、推荐系统等。

本书适合作为高等院校计算机、大数据相关专业的教材，也适合作为软件工程师、应用开发人员、科研人员的参考书。

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码：100037）
责任编辑：谢晓芳　　责任校对：殷　虹
印　　刷：北京市兆成印刷有限责任公司　　版　　次：2018 年 7 月第 1 版第 1 次印刷
开　　本：185mm×260mm　1/16　　印　　张：15.25
书　　号：ISBN 978-7-111-60153-1　　定　　价：69.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzjsj@hzbook.com

# 译者序

随着计算机技术的不断发展和推广，大量的数据随之产生。对这些庞大且不断增长的数据进行存储、处理和分析的需求应运而生。如今，很多高级的 Hadoop 框架（如 Pig、Hive、Scoobi、Scrunch、Cascalog、Scalding 和 Spark）使得 Hadoop 易于操作。本书全面具体地介绍了使用开放源码的工具，如 Hadoop、Scalding、Spark 等来创建和构建分布式处理系统的方法，为读者使用自由及开放源码软件技术（如 Hadoop、Scalding 和 Spark）提供指导和相关实例。

与当下有关的大数据技术书籍相比，本书更具实际应用价值，有着如下鲜明的特色：

第一，从讨论各种形式的分布式系统开始，解析了它们的一般架构，也谈及了其设计的核心，即分布式文件系统。之后，通过相关的示例说明其在发展过程中遇到的技术难题和该领域近年来的发展趋势。

第二，对 Hadoop 生态系统概况进行了讨论，并对系统的安装、编程和实现做了详细说明。书中描述了 Spark 的核心——弹性分布式数据库，并谈及其安装、API 编程和范例。第 4 章重点阐述 Hadoop 流，也涉及了 Scalding 的应用，并讨论了 Python 在 Hadoop 和 Spark 中的应用。

第三，本书并不局限于解释基本的理论常识，它最大的优势在于介绍了程序范例。书中有四类案例解析，内容涉及很多应用领域和计算方法。书中不仅讲述了 K 均值聚类算法的实现，还讲述了使用朴素贝叶斯分类器进行数据分类的问题，进一步阐述了使用 Scalding 和 Spark 的分布式系统中数据挖掘和机器学习的方法，并涉及了回归分析。

在本书的整个翻译过程中，东北林业大学的甄云志老师给予了极大的帮助，并且积极地参与到了翻译工作当中，在此对其表示由衷的感谢。此外，感谢哈尔滨广厦学院的伞颖老师，她对本书的翻译提出了有益的建议，为本书的翻译工作顺利完成提供了积极的帮助。同时感谢机械工业出版社的朱劼编辑，由于她的信任和支持，本书的翻译工作才得以顺利进行。

由于译者水平有限，加之翻译时间紧张，译文中可能存在许多不足，敬请各位同行和广大读者批评指正，欢迎大家将发现的错误或提出的意见与建议发送到邮箱 chong-

yue1979@126.com，我们今后将不断完善本书的译本。本书相关的信息也会在华章网站及译者的微信公众号“大数据与数据科学家”（big_data_scientist）发布。

译者

2018 年 5 月于哈尔滨

# 前　言

过去的二十年中，随着计算机的使用越来越广泛，产生了大量的数据。生产与生活中各类设备和工具的数字化也促进了数据的增长。市场中，对这些庞大且不断增长的数据进行存储、处理和分析的需求应运而生。在硬件层面，每秒进行万亿次浮点运算的高性能计算（HPC）系统可以对庞大的数据进行管理。由于单个计算机无法应对其操作的复杂性，因此 HPC 系统需要在分布式环境中运行。可以通过两种趋势实现万亿次浮点的分布式运算。一种是通过全球网络连接计算机，实现复杂数据的分布式管理。另一种是采用专用的处理器，并集中存放，这样可以缩短机器之间的数据传输时间。这两种趋势正在呈现快速的融合之势，必然会为浩繁的数据处理问题带来更为迅捷和有效的硬件解决方案。

在软件层面，Apache Hadoop 在解决庞大数据的管理问题方面已经是久负盛名。Hadoop 的生态系统包括 Hadoop 分布式文件系统（HDFS）、MapReduce 框架（支持多种数据格式和数据源）、单元测试、对变体和项目进行聚类（如 Pig、Hive 等）。它能够实现包括存储和处理在内的全生命周期的数据管理。Hadoop 的优势在于，它通过分布式模块处理大型数据。它还可以处理非结构化数据，这使其更具吸引力。与 HPC 骨干网结合，Hadoop 可以使处理海量数据的任务变得非常简单。

如今，很多高级的 Hadoop 框架，如 Pig、Hive、Scoobi、Scrunch、Cascalog、Scalding 和 Spark，使得 Hadoop 易于操作。它们中大多数都得到著名企业的支持，如 Yahoo（Pig）、Facebook（Hive）、Cloudera（Scrunch）和 Twitter（Scalding），这说明 Hadoop 在工业领域得到了广泛支持。这些框架使用的是 Hadoop 的基础模块，例如 HDFS 和 MapReduce，但是通过创建一个抽象来隐藏 Hadoop 模块的复杂性，为复杂的数据处理提供了一种简单的方法。这个抽象的一个例证就是 Cascading。许多具体的语言是使用 Cascading 的框架创建的。其中一个实例就是 Twitter 的 Scalding，它用来查询存储在 HDFS 中的大型数据集，如 Twitter 上的推文。

Hadoop 和 Scalding 中的数据存储大多基于磁盘。这一结构因其较长的数据寻道和传输时间影响了运行速率。如果数据从磁盘中读取然后保持在内存中，运行速率会提高数倍。Spark 实现了这一概念，并宣称其效率较之 MapReduce 在内存中快 100 倍，在磁盘上快 10 倍。Spark 使用了弹性分布式数据集的基本抽象，这些数据集是分布式的不可变集合。由于 Spark 将数据存储在内存中，因此迭代算法可以在数据挖掘和机器学

习方面更有效地发挥作用。

## 目标

本书旨在介绍使用自由和开放源码的工具和技术（如 Hadoop、Scalding、Spark 等）构建分布式处理系统的方法，关键目标包括以下几点。

- 使读者掌握当前使用 Hadoop、Scalding 和 Spark 构建高性能分布式计算系统的新发展。
- 为读者提供相关理论的软件框架和实践途径。
- 为学生和实践者使用自由及开放源码软件技术（如 Hadoop、Scalding 和 Spark）提供指导和实例。
- 使读者加深对与高性能分布式计算（HPDC）相关的新兴范式在构建可扩展软件系统以供大规模数据处理方面的理解。

## 本书结构

本书共 8 章，分成两部分，各章内容概述如下。

**第一部分　高性能分布式计算编程基础**

第 1 章阐述构成现代 HPDC 范式（如云计算、网格和集群系统等）主体的分布式系统的基本知识。从讨论各种形式的分布式系统开始，解析它们的通用架构，也谈及其设计的核心，即分布式文件系统。此外，还通过相关的示例说明其在发展过程中遇到的技术难题和该领域近年来的发展趋势。

第 2 章概述 Hadoop 生态系统，一步步地介绍系统的安装、编程和实现。第 3 章描述 Spark 的核心——弹性分布式数据集，谈及其安装、API 编程，并给出一些范例。第 4 章重点阐述 Hadoop 流，也涉及 Scalding 的应用，并讨论 Python 在 Hadoop 和 Spark 中的应用。

**第二部分　使用 Hadoop、Scalding 和 Spark 的案例研究**

本书并不局限于解释基本的理论常识，它的优势在于提供了程序范例。书中给出四个案例，内容涉及很多应用领域和计算方法，足以令怀疑论者变成 Scalding 和 Spark 的信众。第 5 章讲述 K 均值聚类算法的实现，第 6 章讲述使用朴素贝叶斯分类器进行数据分类。第 7 章进一步阐述使用 Scalding 和 Spark 的分布式系统中进行数据挖掘和机器学习的方法，并概述回归分析。

当前，推荐系统在诸多领域都非常受欢迎。它自动充当了两个不相交实体的中间人，在购物、检索、出版领域的现代网络应用中正日趋流行。一个可运行的推荐系统不仅需要有强大的计算引擎，还应该能够实时扩展。第 8 章阐释使用 Scalding 和 Spark 创建这样一个推荐系统的过程。

## 目标受众

本书的目标受众主要包括：

- 软件工程师和应用开发者
- 学生和大学讲师
- 自由和开放源码软件的贡献者
- 研究人员

## 代码库

书中使用的源码和数据集可以从 https://github.com/4ni1/hpdc-scalding-spark 下载。

## 致谢

感谢以下人员在本书的准备过程中提供的支持和帮助：

- M. S. 拉迈阿理工学院董事 M. R. Seetharam 先生
- M. S. 拉迈阿理工学院董事 M. R. Ramaiah 先生
- M. S. 拉迈阿理工学院行政主管 S. M. Acharya 先生
- M. S. 拉迈阿理工学院院长 S. Y. Kulkarni 博士
- M. S. 拉迈阿理工学院副院长 N. V. R. Naidu 博士
- M. S. 拉迈阿理工学院教务主任 T. V. Suresh Kumar 博士

感谢 M. S. 拉迈阿理工学院计算机科学与工程系的所有老师在本书的准备过程中给予我们灵感和鼓励。感谢 P. M. Krishnaraj 先生和 Siddesh G. M. 博士的指导。同样感谢 Nikhil 先生和 Maaz 先生在本书编写上提供及时的帮助。感谢 Scalding 和 Spark 社区给予的支持。

感谢家人的支持与理解。

K. G. Srinivasa

Anil Kumar Muppalla

# 作者简介

## K. G. Srinivasa

K. G. Srinivasa 于 2007 年获得班加罗尔大学计算机科学与工程博士学位。现就职于班加罗尔的 M. S. 拉迈阿理工学院计算机科学与工程系，任教授兼主任。曾荣获印度技术教育委员会青年教师职业奖、印度技术教育学会 ISGITS 全国青年教师优秀研究工作奖、工程师学会（印度）IEI 青年计算机工程师奖、2012 年国际教育技术学会（ISTE）拉贾拉姆巴布·帕蒂尔全国杰出工程教师奖、IMS 新加坡访问科学家奖。他在国际会议和期刊上共发表过一百多篇研究论文，曾作为访问学者出访过许多大学，包括美国俄克拉荷马大学、美国艾奥瓦州立大学、中国香港大学、韩国大学、新加坡国立大学。他撰写的两本书《File Structures Using C++》和《Soft Computer for Data Mining Applications》被收录在 Springer LNAI 系列丛书中。由于与墨尔本大学云实验室在云计算领域开展的合作研究，他获得了 DST 的 BOYSCAST 奖学金。他是 UGC、DRDO 和 DST 资助的多个项目的首席研究员，其研究领域包括数据挖掘、机器学习、高性能计算和云计算。他是 IEEE 和 ACM 的高级成员。可以通过以下邮箱和他取得联系：kgsrinivas@msrit.edu。

## Anil Kumar Muppalla

Anil Kumar Muppalla 先生既是一位研究者也是一位作家，具有计算机科学和工程学学位。他是很多行业的软件开发者和顾问。他是活跃的研究者，并在国际会议和期刊上发表诸多文章。他的研究方向包括使用 Hadoop、Scalding 和 Spark 进行应用开发。他的邮箱是：anil@msrit.edu。

# 目　录

第一部分

# 高性能分布式计算编程基础

# 第 1 章
# 引　言

分布式计算涉及一系列内容和主题。本章总体上阐明分布式系统的一些属性，简要介绍了在成功构建的分布式系统中较为受欢迎的架构，以及所应用的不同类型的系统。并进一步明确了在一些研究领域面临的挑战和获得的启示。本章结尾预测了分布式系统的发展趋势，列举了在某些领域已经做出了突出贡献的实例。

高性能分布式计算（HPDC）适用于需要大量计算机共同执行某项任务的计算活动。其主程序包括数据存储和分析、数据挖掘、仿真建模、科学计算、生物信息、大数据、复杂网络可视化以及更多方面。

早期高性能计算（HPC）系统更多地用于与分布式系统较为相近的并行体系结构的程序运行。现阶段则转移到结构更明晰且利用更有效的分布式计算架构上实现，例如集群、网格和云计算。

随着使用传统计算机语言硬编码方式设计的 HPC 程序越来越不受青睐，像 Hadoop 和 Spark 这样的分布式软件框架应运而生，推动了适用于大规模 HPC 系统的高效程序的发展。像 MapReduce 这样的函数式编程语言模型可以通过 Hadoop 和 Spark 在 HPC 集群上轻易地实现。这类模式的发展很大程度上受到了分布式计算原理的启发。



## 1.1　分布式系统

分布式计算是研究分布式系统部署中多任务计算的计算机科学分支。它是一种计算机网络化布局，各个（计算机）节点间的信息交流通过复杂的消息传递接口来实现。分布式系统主要用来处理那些往往需要几百台计算机协同才能处理和完成的数据集上的问题。有了这样的系统，我们就能在更广泛的领域中解决更多的难题。分布式计算已成为热门的研究领域，且实至名归。

以上只是对分布式系统的一个概述，当然并不全面。而论及分布式系统的物理特性，既然我们称之为分布式系统，那么就涉及系统 I/O 是否与处理器相距较远，或者与处理器相去甚远的存储设备是否在线的问题。无论哪一种情况，仅仅就物理构件分布一方面而言，

它不能完全满足这一方面的要求。目前，能够避免这一矛盾比较认可的系统是从逻辑性和功能性上来定义的分布式系统。逻辑性和功能性通常基于下列标准，当然并不局限于此。

1. **多进程**：系统内不只包含一个顺序进程。这些进程要么是系统指定，要么是用户自定义的，但是每一个进程必须有一个独立的控制线程，无论是外部的还是内部的。

2. **进程通信**：对于分布式系统来说，进程通信信道至关重要，信道的可靠性和信息交互延时取决于某一个节点或者网络布局上连接的物理特性[1]。这里还涉及两个方面：a）内核空间——共享存储、信息传递、信号量等；b）用户空间——后台网络传输、分布式共享存储等。

3. **独立的地址空间**：进程应该具有独立的地址空间。尽管内存共享可以通过通信来完成，但是仅仅是共享内存的多处理器并不符合完全意义上的分布式计算系统的要求[2]。

尽管以上讨论对分布式系统的特性提出了有效标准，但是这些仍然是不够的。比如在分布式布局中，系统进程间通信管理、数据和网络安全同样十分重要。而进程运行时间管理和用户自定义计算控制对于分布式系统特性描述也至关重要。不过，这些内容对于在逻辑上实现分布式计算是完全足够的。系统的物理分布只是逻辑分布的一个先决条件。

计算机和各类网络系统通过网络进行连接。互联网就是一个很好的例子。许多小型网络都与之连接，例如移动电话网络、公司网络系统、制造网络单元、公共机构网络、家庭个人网络，还有公交网络等。这些网络无所不在，并且不断增长，这样的趋势正好满足了许多要突破分布式计算间屏障以形成整体布局的实际情况需求。所有这些网络都具有某些相似的特征，具备成为分布式计算领域研究主体的完整基础条件[3]。图 1-1 是对上述内容的可视化表现。它呈现了这一过程的基本特征，包括网络计算机在哪一个层面进行运算，通过分布式服务或者由程序模仿出的分布式存储和通信接口的软件结构与其他计算机共同进行计算。



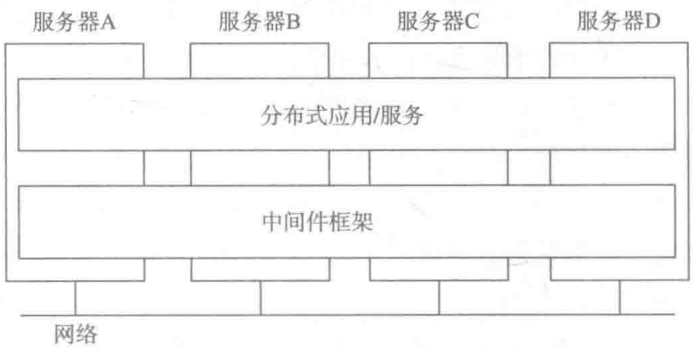



图 1-1 高级分布式系统布局

在分布式系统的构建中，无论分布于各个地区还是建在同一座建筑物内，都要面对以下几个挑战。

1. **程序并发**：在执行分布式系统的过程中，并发程序的执行是一个常态。面临的挑战是实现合理的通信以缩短计算机在使用共享资源的过程中的等待时间。程序的并发执行是在处理诸如数据库一类的事务流程方面最为合适的方式。系统解决共享资源的空间问题

时，可以通过在网络中增加更多资源（如计算机）的方式来扩充。

**2. 缺少中央时钟**：当分布于网络的各个程序需要进行合作的时候，它们会通过消息
5 传递接口交换信息。进程间的密切协调需要一个共享时钟来记录程序的状况和进程。观察发现，由于没有全局时钟的概念，就不能提供精确的参照，以供网络中各节点的计算机同步各自的时间。这一结果正是由于只有一种通信方式，即通过网络发送消息造成的。

**3. 独立故障**：因为基础设施出现故障造成断网的问题最为常见，所以设计者在构建系统时要充分考虑这一因素。计算机系统经常会出现故障，由此出现的问题应及时得到解决。某些故障会造成网络中计算机连接中断，尽管中断连接的计算机仍在运行，但分布式系统将无法运转。同样，如果系统崩溃或某项进程意外终止，且没有向其他与之进行信息交换的组件发出警告，迫使整个运转停滞，系统中的节点也会因此而出现故障。系统中的每一个组件都会在其他组件正常运行的情况下出现故障。这一特性带来的后果需要加以解决，而这一问题的解决方法通常称为容错机制[3]。

创建和维护分布式系统的初衷是实现资源共享。资源是对任何事物的一般性描述，从存储盘、打印机到软件定义的数据库和数据对象，还包括数字流媒体，不论是音频还是视频。在过去几年里，分布式系统的重要性凸显，原因是多方面的，主要体现如下。

**1. 地理分布环境**：分布式系统最显而易见的特点就是计算机遍布全球的物理分布。以银行为例，它们遍布各地以受理客户账户业务。如果能够实现对跨行交易进行监控，对遍布全球的自动提款机资金流向实行监管和记录，并且能够受理全球客户账户业务等，那么银行业就真正实现了互通，也就是所谓的全球化。再比如互联网，它甚至可以把网络运行的最终端变成分布式系统的一部分。另外，网络用户的移动属性进一步为地理分布增加了新的维度。

**2. 计算速度**：或许正是不断提高的计算速度成为人们对分布式结构思考的最主要原因。单处理器在单位时间内的运算量是有物理上限的。而超标量和超长指令字（VLIM）结构体系通过引入指令级并行的概念，强有力地推进了处理能力的提升，但是这两项技术并没有实现质的飞跃。还有一种方法是将多个处理器进行叠列，并且把问题分割成小块，再分配给单个处理器并发执行。这种方法具有一定的扩展性，处理能力可以随着叠加更多的处理器而逐渐提升，比购买一个更高级的处理器更简单而且经济。时下，大数据计算孕
6 育了软件系统的发展，能够根据数据的可分布性，通过将问题分成更小的单元，并借由网络进行传递，来实现多核运算。

**3. 资源共享**：资源既包括软件也包括硬件。分布式系统能遍及各处归因于设备资源的共享。计算机 A 的使用者可能想要使用连接在计算机 B 上的打印机，或者计算机 B 的使用者需要使用计算机 C 硬盘上可供使用的额外存储。同样地，A 工作站可能需要使用 B 工作站和 C 工作站富余的计算能力来提高当前的计算速度。这些事件构成了分布式系统的理想用例。分布式数据库是共享软件资源方面很好的例证，大型文件可以存储在几台主机里，并由一定数量的进程不断进行检索和更新。

在网络资源已成家常便饭的今天，资源共享也已成平常事。像打印机、文档，甚至具

体到功能上，如搜索引擎等硬件资源的共享也习以为常。从硬件资源供给的角度上来说，共享打印机和磁盘一类的资源可以削减成本，但是对于分享，用户的兴趣却来自更高的层面，比如应用程序，还有日常活动，等等。用户更乐于分享网页，而不愿分享个人的硬件设施；同样，他们更热衷于分享搜索引擎和货币转换器这类应用程序，而非它们依托的服务器。这让我们必须意识到，用户之间的合作方式决定了资源共享在不同领域里实现程度的千差万别。例如，搜索引擎向全世界的用户提供服务，而一个封闭的用户群体仅仅通过共享文档进行合作。

4. **容错机制**：针对高性能单一处理器构建的软件很容易在处理器出现故障的情况下崩溃，存在一定的风险。最好是当处理元素的某一个微小的部分出现故障但还有机会实现优雅降级或者改善容差时，做出一点让步，通过使用分布式系统进行处理。这一方法还可以通过利用冗余的处理元素来提高系统的可靠性和可用性。很多时候，系统具有三模冗余结构，三个完全相同的功能单元同时执行相同的操作，以多数输出作为正确输出。在其他容错分布式系统中，处理器在预定义检查点对数据逐一进行交叉校验，自动施行故障检测、故障诊断和故障修复。这样一来，分布式系统完美兼容了容错机制和优雅降级。 7

## 1.2 分布式系统类型

### 1.2.1 分布式嵌入式系统

以上关于分布式系统的叙述大多关于物理分布层面，固定节点通过相对永久的、高质量的方式与网络相连接。大量的现有技术保证了其稳定性，以至于给我们一种印象，认为故障只是偶尔才会发生。像屏蔽和故障恢复这类技术手段可以有效掩盖节点在分布方面的问题，使得使用者或程序相信系统的可靠性。

然而，近年来，随着移动嵌入式计算机设备数量的增长，之前的假设受到了挑战，换而言之，没有那么绝对了。这些设备绝大多数采用电池供电，可移动，并且使用无线网络连接。这类分布式嵌入系统变成我们周围环境的一部分。这些设备缺乏一般管控，至多需要使用者对其进行设置，再或者需要它们通过自动探测，尽可能接入周围合适的网络环境。下面几条判据可以帮助我们认知：

- 根据环境变化做出反应
- 有利于自组网络构建
- 识别共享

根据环境变化做出反应，指的是这些设备必须不停地对周围环境进行识别。其中最浅显的一点就是发现可用无线网络。

自组网络构建，指的是普适系统的各类用例。这样一来，这些设备就必须由用户在设备上运行一套程序或者通过自动处理来完成配置。设备接入网络并获取信息，这就需要一种便于读写、存储、管理和共享信息的方式与方法。信息驻留的地址始终在变化，因此设备必须能够依据提供的服务及时做出反应。数据、进程和操作控制的分布是这类系统的一

方面特性，需要展现，而不是隐藏起来。下面是几个普适系统的具体示例。

- **家庭系统**：家庭网络已经逐渐成为最受欢迎的普适系统。网络中通常至少包括一台计算机，主要把家用电子产品，例如电视机、影音设备、游戏设备、智能手机、掌上电脑，还有其他可穿戴设备连入网络。另外，厨房电器、监控摄像、钟表、
8 照明控制系统等也可以连入一个单独的分布式系统。在家庭网络走进现实之前，还必须着手解决几个问题。自配置和自管理应该是家庭网络系统最重要的两个特征。如果一个或多个设备易出现故障，那么这套系统也就出现故障了。系统的许多问题是通过 UPnP（通用即插即用）规范解决的，在此规范下，家庭设备可以自动获取 IP 地址，彼此间能够识别和交流信息。但是，建立无缝分布式系统，还需要做更多工作。

与分布式家庭网络系统有关的问题还包括：我们还不清楚设备的软件和固件如何在没有人工干预的情况下轻易实现更新。考虑到家庭网络系统存在共享设备和个人设备的布局独特性，我们尚不清楚如何管理个人空间，因为其数据在家庭系统中也属于个人空间问题。人们在这一领域的大部分兴趣都在于建立这部分个人空间。值得庆幸的是，问题可能会变得更简单。家庭系统本身就被设想为分布式的，其分散的特点会导致数据同步的问题。不过，通过提升硬盘容量，同时缩减数据量，这些问题终将迎刃而解。配置 TB 级存储单元已经变得容易，而且尺寸也越来越小，上百 GB 的存储单元开始用于移动设备上。这一功能允许我们建立一个主客户机，用单独一个系统存储和管理网络系统上的所有数据，其他所有客户机只作为进入主系统的接口。这种方法并没有解决如何管理个人空间的问题，存储大量数据的能力将问题转移到存储相关数据并能在以后查找的工作上。近些年来，家居系统配备了推荐功能，它可以从存储的数据中识别相似的信息，并随后导出与用户的个人空间相关的内容。

- **医疗保健**：有一类很重要的普适系统，它基于电子医疗保健的需求。随着医疗费用不断增长，人们研发出新型的设备，用于对个人身体状况进行监控，并自动更新相关医生的资料。这些系统的目标是安置在体域网（BAN）中，它对人的妨碍很小。该网络需要具备无线通信能力，而且保证在人体运动状态下能够正常运行。这就要求它包含很明确的两部分构造，如图 1-2 所示。第一部分是主站，作为 BAN 的一部分，在数据生成时负责收集。收集的数据及时存储在数据暂存器内，主站以此对 BAN 进行管理。第二部分中，BAN 通过无线网络不断传送监控数据。
9 管理 BAN 需要用到不同的技术。
- **传感器网络**：普适系统的最后一个例子是传感器网络。这类网络很多情况下构成了许多普适应用的实现技术。就分布式系统而言，传感器网络之所以成为关注的兴趣点，是因为它们几乎全部用来处理信息。

传感器网络通常包含几十到几百或者几千个节点。每一个节点都装配了传感器。大多数传感器网络都是用无线网络通信，节点通常由电池供电。因为自身资源不足，通信能力有限而且受到能耗制约，所以在设计标准上要求很高。