<<UNIX网络编程：第2版. 第2卷， >>

图书基本信息

书名：<<UNIX网络编程：第2版. 第2卷， 进程间通信(中文版)>>

13位ISBN编号：9787115230287

10位ISBN编号：7115230285

出版时间：2010-7

出版时间：人民邮电出版社

作者：(美)W. Richard Stevens

页数：454

前言

大多数重要的程序都涉及进程间通信（Inter process Communication，IPC）。
这是受下述设计原则影响的自然结果：把应用程序设计为一组相互通信的小片断比将其设计为单个庞大的程序更好。
从历史角度看，应用程序有如下几种构建方法。
（1）用一个庞大的程序完成全部工作。
程序的各部分可以实现为函数，函数之间通过参数、返回值和全局变量来交换信息。
（2）使用多个程序，程序之间用某种形式的IPC进行通信。
许多标准的Unix工具都是按这种风格设计的，它们使用shell管道（IPC的一种形式）在程序之间传递信息。
（3）使用一个包含多个线程的程序，线程之间使用某种IPC。
这里仍然使用术语IPC，尽管通信是在线程之间而不是在进程之间进行的。
还可以把后两种设计形式结合起来：用多个进程来实现，其中每个进程包含几个线程。
在这种情况下，进程内部的线程之间可以通信，不同的进程之间也可以通信。
上面讲述了可以把完成给定任务所需的工作分到多个进程中，或许还可以进一步分到进程内的多个线程中。
在包含多个处理器（CPU）的系统中，多个进程也许可以（在不同的CPU上）同时运行，或许给定进程内的多个线程也能同时运行。
因此，把任务分到多个进程或线程中有望减少完成指定任务的时间。

内容概要

本书是一部UNIX网络编程的经典之作！
进程间通信(IPC)几乎是所有Unix程序性能的关键，理解IPC也是理解如何开发不同主机间网络应用程序的必要条件。
本书从对Posix IPC和System V IPC的内部结构开始讨论，全面深入地介绍了4种IPC形式：消息传递(管道、FIFO、消息队列)、同步(互斥锁、条件变量、读写锁、文件与记录锁、信号量)、共享内存(匿名共享内存、具名共享内存)及远程过程调用(Solaris门、Sun RPC)。
附录中给出了测量各种IPC形式性能的方法。
本书内容详尽且具权威性，几乎每章都提供精选的习题，并提供了部分习题的答案，是网络研究和开发人员理想的参考书。

作者简介

作者：（美国）W.Richard StevensW.Richard Stevens，国际知名的UNIX和网络专家，备受赞誉的技术作家他1951年2月5日出生于赞比亚，后随父母回到美国中学时就读于弗吉尼亚菲什伯恩军事学校，1973年获得密歇根大学航空和航天工程学士学位，1975年至1982年，他在亚利桑那州图森市的基特峰国家天文台从事计算机编程工作，业余时间喜爱飞行运动，做过兼职飞行教练这期间他分别在1978年和1982年获得亚利桑那大学系统工程硕士和博士学位此后他去康涅狄格州纽黑文的健康系统国际公司任主管计算机服务的副总裁，1990年他回到图森，从事专业技术写作和咨询工作写下了多种经典的传世之作。

<<UNIX网络编程：第2版. 第2卷，>>

书籍目录

章节摘录

插图：即使一个进程终止时系统会自动释放某个锁，那也可能解决不了问题。
该锁保护某个临界区很可能是为了在执行该临界区代码期间更新某个数据。
如果该进程在执行该临界区的中途终止，该数据处于什么状态呢？
该数据处于不一致状态的可能性很大：举例来说，一个新条目也许只是部分插入某个链表中，要是该进程终止时内核仅仅把那个锁解开的话，使用该链表的下一个进程就可能发现它已损坏。
然而在某些例子中，让内核在进程终止时清理某个锁（若是信号量情况则为计数器）不成问题。
例如，某个服务器可能使用一个SystemV信号量（打开其SEM-UNDO特性）来统计当前被处理的客户数。
每次fork一个子进程时，它就把该信号量加1，当该子进程终止时，它再把该信号量减1。
如果该子进程非正常终止，内核仍会把该计数器减1。
9.7节给出了一个例子，说明内核在什么时候释放一个锁（不是我们刚讲的计数器）合适。
那儿的守护进程一开始就在自己的某个数据文件上获得一个写入锁，然后在其运行期间一直持有该锁。
如果有人试图启动该守护进程的另一个副本，那么新的副本将因为无法取得该写入锁而终止，从而确保该守护进程只有一个副本在一直运行。
但是如果该守护进程不正常地终止了，那么内核会释放该写入锁，从而允许启动该守护进程的另一个副本。

后记

本书详细讲述了用于进程间通信（IPC）的四种不同技术：（1）消息传递（管道、FIFO、Posix
莉lSystemV消息队列）；（2）同步（互斥锁、条件变量、读写锁、文件和记录锁、.Posix和SystemV信号量）；（3）共享内存区（匿名共享内存区、有名Posix共享内存区、有名SystemV共享内存区）；（4）过程调用（Solaris门、SunRPC）。
消息队列和过程调用往往单独使用，也就是说它们通常提供了自己的同步机制。
相反，共享内存区通常需要某种由应用程序提供的同步形式才能正确工作.同步技术有时候单独使用，也就是说不涉及其他形式的IPC。
讨论了共16章的细节后，很显然的一个问题是：解决某个特定问题应使用哪种形式的IPC？
遗憾的是不存在关于IPC的简单判定。
Unix提供的类型如此之多的IPC表明，不存在解决全部（或者甚至于大部分）问题的单一办法。
你能做的仅仅是：逐渐熟悉各种IPC形式提供的机制，然后根据特定应用的要求比较它们的特性。
我们首先列出必须考虑的四个前提，因为它们对于你的应用程序相当重要。
（1）连网的（networked）还是非连网的（nonnetworked）。
我们假设已作出这个决定，IPC就是用于单台主机上的进程或线程间的。
如果应用程序有可能散布到多台主机上，那就考虑使用套接字代替IPC，从而简化以后向连网的应用程序转移的工作。

编辑推荐

《UNIX网络编程.卷2:进程间通信(第2版)》：图灵计算机科学丛书

版权说明