



嵌入式Linux 应用编程

许信顺 贾智平 编著



机械工业出版社
China Machine Press



中国电子学会嵌入式专家委员会指定教材

嵌入式Linux

应用编程

TP316.81

112

2007

许信顺 贾智平 编著



机械工业出版社
China Machine Press

本书对嵌入式Linux系统进行了详细、系统的介绍,并对嵌入式Linux环境下的应用程序开发知识进行了介绍,包括嵌入式Linux基本概念和开发流程、BootLoader的工作原理、内核裁减配置和交叉编译、根文件系统制作、网络编程及图形界面开发等知识。

本书内容丰富,叙述清楚,大部分章节配有实例和程序,可作为嵌入式Linux认证考试的教材;同时,也是从事嵌入式编程开发人员的好工具。

版权所有,侵权必究。

本书法律顾问 北京市展达律师事务所

图书在版编目(CIP)数据

嵌入式Linux应用编程 / 许信顺等编著. — 北京:机械工业出版社, 2007.1
ISBN 7-111-20483-2

I. 嵌… II. 许… III. Linux操作系统—教材 IV. TP316.89

中国版本图书馆CIP数据核字(2006)第147649号

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:杨庆燕

北京牛山世兴印刷厂印刷·新华书店北京发行所发行

2007年1月第1版第1次印刷

186mm×240mm·13.5印张

定价:29.00元

凡购本书,如有倒页、脱页、缺页,由本社发行部调换
本社购书热线:(010) 68326294

专家指导委员会

(按姓氏笔画顺序)

主 任：王 越 (院士)

副主任：王田苗 (教授) 魏洪兴

委 员：马洪连 王 磊 田景文 关 永
张国印 闵华松 严 义 邵子立
邱秀芹 周亦敏 杨 刚 贺 平
贾智平 康一梅 章登义 潘巨龙
魏洪兴 魏淑桃

前 言

随着科学技术的发展，嵌入式系统已经渗透到各个应用领域，例如工业控制、国防军事、医疗卫生和消费娱乐等。人们在日常生活中所使用的各种电子设备，比如手机、掌上电脑、PDA等，均含有嵌入式系统。除了以上我们比较熟悉的设备以外，还有更多的嵌入式系统隐身于不为人知的角落，小到电子手表、电子词典、电子体温计，大到空调、电冰箱、汽车的导航系统、微电脑系统、医疗器材以及工业自动控制等。可以说在我们的生活中嵌入式系统无处不在，已经成为我们生活中不可缺少的一部分。所以，嵌入式系统是继IT网络技术之后，又一个新的技术发展方向，它对社会的各个领域产生了越来越深远的影响。

嵌入式系统在其发展过程中，大体经历了以下四个阶段：无操作系统阶段，简单操作系统阶段，实时操作系统阶段和面向Internet阶段。在嵌入式系统的不同发展阶段，对于嵌入式操作系统功能上的要求也逐渐由简单向复杂过渡。同时，因为新的微处理器层出不穷，嵌入式操作系统自身结构的设计必须要更加便于移植，要求在短时间内支持更多的微处理器。尤其，随着网络化、信息化的发展，随着Internet技术的成熟和带宽的提高，网络互联成为必然趋势，以往功能单一的设备如电话、手机、冰箱、微波炉等功能不再单一，从而对嵌入式操作系统提出了更高的要求。比如，对于面向Internet的嵌入式系统，不但要求其嵌入式操作系统具备文件管理、设备管理、多任务管理、网络连接和图形用户界面（GUI）等功能，而且要求尽量提供较多的应用程序接口（API），从而便于应用软件的开发。

目前，市场上流行的嵌入式操作系统有许多，比如Windows CE、Palm OS、VxWorks、uClinux和嵌入式Linux系统等，这些嵌入式操作系统在开放性、实用性以及性能等方面各有千秋。其中，由于嵌入式Linux操作系统具有源代码开放、系统内核小、执行效率高、网络结构完整等特点，使其很适合各类嵌入式系统的需求，所以嵌入式Linux操作系统已经成为Windows CE、Palm OS等嵌入式操作系统的有力竞争对手。随着嵌入式Linux系统在各种嵌入式系统中的应用，在嵌入式Linux环境下的编程也逐渐成为相关研究、开发人员讨论的热点。如果有一本关于嵌入式Linux环境下的编程书籍，无疑对从事该方面工作或将从事这方面工作的人员起到抛砖引玉的作用。这也正是我们编写本书的初衷。

本书分为10章：第1章为Linux操作系统入门基础；第2章介绍了Linux系统下的程序设计基础，尤其对C环境进行了介绍；第3章介绍了Linux下的shell编程；第4章就如何构建嵌入式Linux系统进行了叙述；第5章对嵌入式BootLoader技术进行了详细的分析；第6章以嵌入式操作系统ARM-Linux为例，对其内核进行了详细的剖析；第7章对嵌入式系统中的文件系统进行了综合描述和具体分析；第8章介绍了如何在嵌入式Linux环境下进行多线程编程；第9章就构建嵌入式Web服务器使用的相关技术进行了详细的分析介绍；最后，在第10章中，我们介绍了嵌入式GUI。

本书由魏洪兴策划，许信顺主编，贾智平对本书进行了审校。参加编写工作的还有王通、刘甜甜、王谓中、程鹏、王宁。北京航空航天大学许多老师对本书编写给予了大力支持和帮助，在此表示感谢。

本书参考了《Building Embedded Linux Systems》、《TCP/IP Lean Web Servers for Embedded Systems》等书籍。同时，书中有许多内容取材于国内外最新的技术资料。书末有具体的参考文献，有兴趣的读者可以进一步查阅。

限于编者的水平，加上嵌入式技术还在不断发展，有许多问题有待进一步探讨，书中可能存在谬误和不妥之处，恳请读者批评指正。

许信顺

2006年秋于山东大学

目 录

专家指导委员会

前言

第1章 Linux系统入门基础1

1.1 认识Linux操作系统1

1.1.1 Linux和Unix的简明历史1

1.1.2 Linux系统的特点和组成3

1.1.3 Linux的开发过程4

1.2 基本操作命令6

1.2.1 字符界面简介6

1.2.2 常用命令简介6

1.3 Linux文件与目录系统11

1.3.1 Linux文件系统类型介绍11

1.3.2 Linux目录系统14

1.4 shell简介16

1.5 网络服务简介18

1.5.1 Linux支持的网络协议18

1.5.2 Linux的网络服务19

第2章 Linux程序设计基础——C环境21

2.1 Linux编程环境介绍21

2.1.1 系统平台环境21

2.1.2 开发工具环境21

2.1.3 基于文本模式的开发平台22

2.1.4 集成开发平台Eclipse+CDT23

2.1.5 文档帮助环境24

2.2 常用编辑器25

2.2.1 VIM编辑器26

2.2.2 Emacs编辑器29

2.3 gcc编译器的使用32

2.3.1 gcc的主要选项33

2.3.2 GNU C扩展简介36

2.4 GNU make管理项目37

2.4.1 make简介37

2.4.2 编写Makefile文件的规则38

2.5 GDB调试42

2.5.1 GDB命令介绍42

2.5.2 GDB调试例程46

2.5.3 基于GDB的图形界面调试工具49

第3章 Linux shell编程51

3.1 创建和运行外壳程序51

3.1.1 创建外壳程序51

3.1.2 运行外壳程序51

3.2 使用外壳变量52

3.2.1 给变量赋值52

3.2.2 读取变量的值53

3.2.3 位置变量和其他系统变量53

3.2.4 引号的作用54

3.3 数值运算命令55

3.4 条件表达式57

3.4.1 if 表达式57

3.4.2 case 表达式58

3.5 循环语句60

3.5.1 for语句60

3.5.2 while语句61

3.5.3 until语句62

3.6 shift命令63

3.7 select语句63

3.8 repeat语句64

3.9 函数64

3.10 shell应用举例67

第4章 构建嵌入式Linux系统71

4.1 GNU跨平台开发工具链71

4.1.1 基础知识	71	6.5 ARM-Linux系统启动与初始化	123
4.1.2 GNU跨平台开发链的建立过程	71	6.5.1 内核数据结构的初始化	123
4.2 嵌入式Linux内核	78	6.5.2 外设初始化	126
4.3 嵌入式Linux根文件系统	79	6.5.3 init进程和inittab文件	126
4.4 BootLoader简介	80	6.5.4 rc启动脚本	127
第5章 嵌入式BootLoader技术	83	6.5.5 shell的启动	128
5.1 BootLoader的基本概念	83	6.6 Linux模块化机制	128
5.1.1 BootLoader所支持的CPU 和嵌入式板	83	6.6.1 Linux的模块化	128
5.1.2 BootLoader的安装点和启动过程	84	6.6.2 模块的载入	129
5.1.3 BootLoader的操作模式	84	6.6.3 模块的卸载	130
5.1.4 BootLoader与主机之间的通信方式	85	6.7 ARM-Linux内核裁减	130
5.2 BootLoader典型结构框架	85	第7章 嵌入式文件系统	133
5.2.1 BootLoader的stage1	86	7.1 Linux文件系统结构与特征	133
5.2.2 BootLoader的stage2	88	7.1.1 Linux文件系统概述	133
5.3 典型BootLoader (Blob) 分析和移植	94	7.1.2 Linux文件系统布局	133
5.3.1 Blob简介	94	7.1.3 Linux支持的文件系统	136
5.3.2 Blob的运行过程分析	95	7.2 嵌入式文件系统简介	137
5.3.3 Blob的移植	97	7.2.1 嵌入式文件系统JFFS简介	138
第6章 ARM-Linux内核分析	100	7.2.2 嵌入式文件系统YAFFS简介	139
6.1 ARM微处理器	100	7.3 构建根文件系统	140
6.1.1 RISC体系结构	100	7.4 根文件系统设置	142
6.1.2 ARM微处理器的寄存器结构	101	7.5 BusyBox	144
6.1.3 ARM微处理器的指令结构	102	7.5.1 BusyBox简介	144
6.2 ARM-Linux的内存管理	103	7.5.2 使用BusyBox构建根文件系统	146
6.2.1 内存管理单元MMU	103	第8章 嵌入式Linux多线程编程	149
6.2.2 ARM-Linux的存储管理机制	103	8.1 线程基本概念	149
6.2.3 ARM-Linux存储机制的建立	105	8.1.1 Linux线程简介	149
6.2.4 ARM-Linux对进程虚拟空间的管理	107	8.1.2 Linux线程编程基础	149
6.3 ARM-Linux进程管理与调度	109	8.2 多线程同步	154
6.3.1 task_struct数据结构	109	8.2.1 互斥锁	154
6.3.2 Linux进程的创建、执行和消亡	113	8.2.2 条件变量	156
6.3.3 ARM-Linux的进程调度	117	8.2.3 信号量	158
6.4 ARM-Linux中断响应与处理	118	8.3 生产者-消费者问题	161
6.4.1 ARM的异常中断种类	118	8.3.1 生产者-消费者问题简介	161
6.4.2 ARM处理器对异常中断的响应 及返回过程	119	8.3.2 生产者-消费者问题实例	162
		第9章 简单嵌入式Web服务器设计	167
		9.1 HTTP协议基础	167

9.1.1 HTTP协议概述	167	10.1.1 MiniGUI	182
9.1.2 HTTP协议的方法	168	10.1.2 Qt/Embedded	183
9.1.3 HTTP协议消息头	169	10.1.3 Nano-X	184
9.1.4 HTTP协议使用举例	171	10.1.4 OpenGUI	185
9.2 socket编程基础	172	10.2 MiniGUI程序设计基础	185
9.2.1 socket描述	172	10.2.1 MiniGUI的特点	186
9.2.2 TCP/IP模型	172	10.2.2 窗口处理过程	188
9.2.3 socket描述符	173	10.2.3 MiniGUI的线程机制	189
9.3 嵌入式Web服务器系统结构	174	10.2.4 MiniGUI的体系结构	190
9.3.1 协议标准	174	10.2.5 MiniGUI的底层引擎	191
9.3.2 瘦Web服务器	174	10.2.6 MiniGUI的三种运行模式	192
9.3.3 嵌入式应用接口技术	175	10.3 Qt/Embedded程序设计基础	193
9.3.4 实现形式	176	10.3.1 Qt/Embedded的图形引擎实现基础	195
9.3.5 EMIT	176	10.3.2 Qt/Embedded的事件驱动基础	197
9.3.6 RomPager	177	10.3.3 Qt/Embedded底层支持分析	197
9.4 Web服务器构建	177	10.4 嵌入式GUI的移植	198
第10章 嵌入式GUI	182	参考文献	205
10.1 嵌入式GUI简介	182		

第1章 Linux系统入门基础

1.1 认识Linux操作系统

让用户很详细地了解现有操作系统的实际工作方式几乎是不可能的，因为大多数操作系统的源代码都是严格保密的，比如Windows和Unix操作系统。当然也有例外，比如有一些操作系统是为研究和教学而设计的，这类系统的源代码是开放式。但是，这类系统的功能往往不能跟实际使用的操作系统相比，只不过是实现了实际操作系统的小部分功能来体现操作系统的某些特性。

在以实际使用为目的的操作系统中，让所有人都可以自由获取系统源代码，无论其目的是要了解、学习还是改进，这样的操作系统并不多。Linux就是这类少数操作系统中的一个。

1.1.1 Linux和Unix的简明历史

由于Linux是在Unix的基础上发展而来的，所以，我们在具体讨论Linux之前，先来看看Unix的发展历程。

Unix是由AT&T贝尔实验室的Ken Thompson和Dennis Ritchie于1969年在一台已经废弃了的PDP-7上开发的；它最初是一个用汇编语言写成的单用户操作系统。不久，Thompson和Ritchie成功地说服管理部门为他们购买更新的机器，以便该开发小组可以实现一个文本处理系统，于是Unix就在PDP-11上用C语言重新编写（发明C语言的部分目的就在于此）。

最终，他们实现了该文本处理工具，而且Unix（以及Unix上运行的工具）也在AT&T得到广泛应用。1973年，Thompson和Ritchie在一个关于操作系统的会议上就这个系统发表了一篇论文，该论文引起了学术界对Unix系统的极大兴趣。

尽管受美国反垄断法案的限制，AT&T不能涉足计算机业务，但AT&T仍然被允许以象征性的费用发售该系统。随后，Unix被广泛发布，它的用户包括学术科研用户，也包括政府和商业用户。

在众多用户中，加州大学伯克利（Berkeley）分校是众多学术用户之一。在这里，Unix得到了计算机系统研究小组（CSRG）的广泛应用。并且在这里所进行的修改也使Unix产生了新的版本，这就是广为人知的BSD Unix。除了AT&T所提供的Unix系列之外，BSD是最有影响力的Unix系列。BSD在Unix中增加了许多显著特性，例如TCP/IP协议，更好的用户文件系统（UFS）工作控制，并且改进了AT&T的内存管理代码。

多年以来，BSD版本的Unix一直在学术界占据主导地位，但最终发展成为System V版本的AT&T的Unix则成为商业领域的主宰。从某种程度上来说，这是有社会原因的：学校倾向于使用非正式但通常更好用的BSD风格Unix，而商业界则倾向于从AT&T获取Unix。

在用户需求驱动和用户编程改进特性的促进下，BSD风格的Unix一般要比AT&T的Unix更具

有创新性，而且改进也更为迅速。但是，在AT&T发布最后一个正式版本System V Release 4 (SVR4) 时，System V Unix已经吸收了BSD的大多数重要的优点，并且还增加了一些自己的优势。这种现象的部分原因在于从1984年开始，AT&T逐渐将Unix商业化，而BSD Unix的开发工作在1993年BSD4.4版本完成后就逐渐收缩以至终止了。但BSD的进一步改进由外界开发者延续下来，直到今天还在继续进行。正在进行的Unix系列开发中至少有四个独立的版本直接起源于BSD4.4，不包括几个厂商的Unix版本，例如惠普的HP-UX，它们的Unix版本都是部分地或者全部地基于BSD发展起来的。

实际上Unix的变种并不止BSD和System V。由于Unix主要使用C语言来编写，这就使得它相对比较容易移植到新的机器上，它的简单性也使其相对比较容易重新设计与开发。Unix的这些特点受到商业界硬件供应商的欢迎，比如Sun、SGI、惠普、IBM、DEC、Amdahl等，IBM也不止一次的对Unix进行了再开发。厂商们设计开发出新的硬件并简单地将Unix移植到新的硬件上，这样新的硬件一经发布便具备一定的功能。经过一段时间之后，这些厂商都拥有了自己的专有Unix版本。而且为了占有市场，这些版本故意以不同的侧重点发布出来以更好地占有用户。

版本混乱的状态促进了标准化工作的进行。其中最主要的是POSIX系列标准，它定义了一套标准的操作系统接口和工具。从理论上说，POSIX标准代码很容易移植到任何遵守POSIX标准的操作系统中，而且严格的POSIX测试已经把这种理论上的可移植性转化为现实。直到今天，几乎所有的正式操作系统都以支持POSIX标准为目标。

现在让我们回顾一下Linux的历史。1984年，杰出的电脑黑客Richard Stallman独立开发出一个类Unix的操作系统，该操作系统具有完整的内核、开发工具和终端用户应用程序。在GNU计划的配合下，Stallman开发这个产品有自己的技术理想：他想开发出一个质量高而且自由的操作系统。Stallman使用了“自由”(free)这个词，不仅意味着用户可以免费的获取软件，而且更重要的是，它将意味着某种程度的“解放”：用户可以自由使用、拷贝、查询、重用、修改甚至是分发这份软件，完全没有软件使用协议的限制。这也正是Stallman创建自由软件基金会(FSF)资助GNU软件开发的本意(FSF也在资助其他科研方面的开发工作)。

15年以来，GNU工程已经吸收、产生了大量的程序，这不仅包括Emacs，gcc (GNU的C编译器)，bash (shell命令)，还有大部分Linux用户所熟知的许多应用程序。现在正在进行开发的项目是GNU Hurd内核，这是GNU操作系统的最后一个主要部件。与现有的Linux操作系统相比，Hurd内核具有以下优点：一是Hurd的体系结构十分清晰的体现了Stallman关于操作系统工作方式的思想，例如，在运行期间，任何用户都可以部分的改变或替换Hurd (这种替换不是对每个用户都可见，而是只对申请修改的用户可见，而且还必须符合安全规范)。二是Hurd对于多处理器的支持比Linux本身的内核要好。

在GNU发展的中期，也就是1991年，一个名叫Linus Torvalds的芬兰大学生想要了解Intel的新CPU——80386。他认为比较好的学习方法是自己编写一个操作系统的内核。出于这种目的，加上他对当时Unix变种版本对于80386类机器的脆弱支持十分不满，他决定要开发出一个全功能的、支持POSIX标准的、类Unix的操作系统内核，该系统吸收了BSD和System V的优点，同时摒弃了它们的缺点。Linus独立把这个内核开发到0.02版，这个版本已经可以运行gcc，bash和很

少的一些应用程序。这些就是他开始的全部工作了。后来，他把独立开发的系统发布到网上，以供人下载、从因特网络上寻求广泛的帮助。

不到三年，Linux的Unix-Linux已经升级到1.0版本。其源代码量也呈指数形式增长，实现了基本的TCP/IP功能（网络部分的代码后来重写过，而且还可能会再次重写）。此时Linux已经拥有大约10万用户了。

现在的Linux内核由150多万行代码组成，Linux也已经拥有了大约1000万用户（由于Linux可以自由获取和拷贝，获取具体的统计数字是不可能的）。Linux内核GNU/Linux附同GNU工具已经占据了Unix 50%的市场。一些公司正在把内核和一些应用程序、安装软件打包在一起，生产出Linux的distribution（发行版本），这些公司包括Red Hat和Calera prominent公司。现在GNU/Linux已经备受瞩目，得到了Sun、IBM、SGI等公司的广泛支持。最近SGI决定在其基于Intel的Merced的系列机器上不再搭载自己的Unix变种版本IRIX，而是直接采用GNU/Linux。

1.1.2 Linux系统的特点和组成

1. Linux系统的特点

Linux操作系统在短短几年内得到了迅猛的发展，这与Linux具有的良好特性是分不开的。简单地讲，Linux具有以下主要特性：

- **开放性**：是指系统遵循世界标准规范，特别是遵循开放系统互联（OSI）国际标准。凡遵循国际标准所开发的硬件和软件，都能彼此兼容，可以方便地实现互联。另外，Linux是源代码开放且是免费的。
- **多用户多任务**：多用户是指系统资源可以被不同用户各自拥有使用。多任务是现代计算机的一个主要特点，是指计算机同时执行多个程序，且各程序相互独立运行。
- **出色的速度性能**：Linux可以连续运行数月数年而无需重启，与NT（经常死机）相比，这项性能尤其突出。
- **良好的用户界面**：Linux向用户提供了三种界面：用户命令界面、系统调用界面和图形用户界面。
- **提供了丰富的网络功能**：Linux是在Internet的基础上发展起来的，因此完善的内置网络协议是Linux的一大特点。Linux在通信和网络功能方面优于其他操作系统。
- **可靠的系统安全**：Linux采取了许多安全技术措施，包括对读写进行权限控制、带保护的子系统、审计跟踪等。
- **良好的可移植性**：可移植性是指将操作系统从一个平台转移到另一个平台时，它仍能按其自身方式运行的能力。Linux是一种可移植的操作系统，能够在从微型机到大型计算机的任何环境和平台上运行。可移植性为运行Linux的不同计算机平台与其他任何机器进行准确有效的通信提供了手段，不需要另外增加特殊、昂贵的通信接口。
- **具有标准兼容性**：Linux是一个与POSIX（Portable Operating System Interface）相兼容的操作系统，它所构成的子系统支持所有相关的ANSI、ISO、IETF和W3C业界标准。为了使Unix System V和BSD上的程序能直接在Linux上运行，Linux还增加了部分System V和BSD的系统接口，使Linux成为一个完善的Unix程序开发系统。另外，Linux在对工业标准的支持上也做得很好。

2. Linux系统的组成

Linux一般有4个部分组成：内核、shell、文件系统和应用程序。内核、Shell和文件系统一起构成了基本的操作系统结构，它们使用户可以运行程序、管理文件并使用系统。

- **Linux内核**：内核是一个操作系统最基本的组成部分，由它来向应用程序访问硬件时提供服务。
- **Linux Shell**：Shell是系统的用户界面，提供用户与内核的交互接口。实际上Shell是一个命令解释器，它接收并解释用户输入的命令并把它们送到内核。
- **Linux文件系统**：文件系统是文件存放在磁盘等存储设备上的组织方法。Linux支持多种目前流行的文件系统，如EXT2、EXT3、FAT、VFAT、ISO9660、NFS、SMB等。
- **Linux应用程序**：标准Linux系统都有一套称为应用程序的程序集，包括文本编辑器、编程语言、X Window、办公套件、Internet工具、数据库等。

1.1.3 Linux的开发过程

如上所述，Linux是一款自由软件，它可以免费获取以供学习研究。但是，Linux之所以受到广大用户的青睐，值得我们去学习和研究，并不单是因为它是免费的操作系统，更主要的是因为它是一款相当优秀的操作系统。它的优秀性主要体现在以下几个方面：

首先，Linux是基于天才的思想开发而成的。在学生时代就开始推动整个系统开发的Linus Torvads是一个天才，他的才能不仅展现在编程能力方面，而且组织技巧也相当杰出。Linux的内核是由世界上一些最优秀的程序员开发并不断完善的，他们通过Internet相互协作，开发理想的操作系统；他们享受着工作中的乐趣，而且也获得了充分的自豪感。

其次，Linux系统中包含了一组优秀的概念。Unix是一个简单却非常优秀的模型。在Linux创建之前，Unix已经有20年的发展历史。Linux从Unix的各个流派中不断吸取成功经验，模仿Unix的优点，抛弃Unix的缺点。这样做的结果是Linux成为了Unix系列中的佼佼者：高速、健壮、完整，而且抛弃了历史包袱。

最后，Linux开发过程是公开的。这也是Linux最强大的生命力之所在。每个人都可以自由获取内核源程序，每个人都可以对源程序加以修改，而后他人也可以自由获取你修改后的源程序。如果你发现了缺陷（bug），你可以对它进行修正，而不用去乞求不知名的公司来为你修正。如果你有什么最优化或者新特点的创意，你也可以直接在系统中增加功能，而不用向操作系统供应商解释你的想法，指望他们将来会增加相应的功能。当发现一个安全漏洞后，你可以通过编程来弥补这个漏洞，而不用关闭系统直到你的供应商为你提供修补程序。由于你拥有直接访问源代码的能力，你也可以直接阅读代码来寻找缺陷，或是效率不高的代码，或是安全漏洞，以防患于未然。

除非你是一个程序员，否则这一点听起来仿佛没有多少吸引力。但是，即使你不是程序员，这种开发模型也将使你受益匪浅，这主要体现在以下两个方面：

- 可以间接受益于世界各地成千上万的程序员随时进行的改进工作。
- 如果需要对系统进行修改，可以雇用程序员为你完成工作。这部分人将根据你的需求定义单独为你服务。

Linux这种独特的自由流畅的开发模型已被命名为bazaar（集市模型），它是相对于cathedral（教堂）模型而言的。Bazaar开发模型非常重视实验，它通过征集并充分利用早期的实验反馈，对巨大数量的脑力资源进行平衡配置，从而开发出更优秀、更完善的软件系统。而在cathedral模型中，源程序代码往往被锁定在一个保密的小范围内。只有开发者（很多情况下是市场）认为能够发行一个新版本，这个新版本才会被推向市场。这些术语在Eric S. Raymond的The Cathedral and the Bazaar一文中有所介绍，大家可以在<http://www.tuxedo.org/~esr/writings/>找到这篇文章。

从以上我们看到Bazaar开发模型是一种基于无序的开发模型，为了确保这些无序的开发过程能够有序地进行，Linux采用了双树系统。一个树是稳定树（stable tree），另一个树是非稳定树（unstable tree）或者开发树（development tree）。一些新特性、实验性改进等都将首先在开发树中进行。如果在开发树中所做的改进也可以应用于稳定树，那么在开发树中经过测试以后，在稳定树中将进行相同的改进。按照Linus的观点，一旦开发树经过了足够的发展，开发树就会成为新的稳定树，如此周而复始的进行下去。

Linux的版本号分为两个部分，即内核（kernel）版本与发行套件（distribution）版本。

1. Linux的内核版本

内核版本是在Linus领导下的开发小组开发出的系统内核版本。

内核版本号由3个数字组成：r.x.y。

r：目前发布的Kernel主版本。

x：偶数：稳定版本；奇数：开发中版本。

y：错误修补的次数。

一般来说，x位为偶数的版本是一个可以使用的稳定版本，如2.4.4；x位为奇数的版本一般加入了一些新的内容，所以性能不一定很稳定，仍是处于测试阶段，系统是一个测试版本，如2.5.5。

2. Linux的发行版本

发行版本是一些组织或厂家将Linux系统内核与应用软件和文档包装起来，并提供一些安装界面和系统设定管理工具的一个软件包的集合，表1-1所示的是常见的Linux发行版本。相对于内核版本，发行套件的版本号随发布者的不同而不同，它与系统内核的版本号是相对独立的。

表1-1 常见的Linux发行版本

	发 行 版 本	公 司 网 址
国际发布与国内发布	Red Hat Linux	http://www.redhat.com/
	Mandrake Linux	http://www.linux-mandrake.com/en/
	SUSE Linux	http://www.suse.com/
	Debian Linux	http://www.debian.org/
	Caldera Linux	http://www.caldera.com/
	Redflag Linux	http://www.redflag.com.cn/
安全发布与小型发布	Astaro Security Linux	http://www.astaro.org/
	Engarde Secrue Linux	http://www.engardelinux.org/
	ClarkConnect	http://www.clarkconnect.org/
	Linux Router Project	http://www.linuxrouter.org/

<http://www.kernel.org>及其镜像站点提供了最新的可供下载的内核版本，而且同时包括稳定和开发版本。

1.2 基本操作命令

1.2.1 字符界面简介

Linux是一种类Unix操作系统。在Unix发展早期，类Unix操作系统根本没有图形界面，只有字符工作模式。后来随着GUI的发展，在类Unix操作系统上开发了X Window系统，使类Unix系统有了图形界面。虽然近几年个人计算机的潮流已经从命令行方式转为图形界面方式，但是shell在Linux中依然有强劲的生命力。shell有很多种，如：bash、ksh、tcsh、zsh、ash，用得最多的还是bash，它几乎是各种Linux发布版的标准配置。同时，即使在X Window下，系统管理员也要经常与命令行打交道。而且应当注意Linux的命令（也包括文件名等）对大小写是敏感的。

当然，我们可以在图形环境下开启终端或系统启动后直接进入字符工作方式，或使用远程登录方式（Telnet或SSH）进入字符工作方式。

1.2.2 常用命令简介

进入系统后可以点击右键打开终端，进入终端后会看到：`[root@tty/]#`，其中第一个root表示登录用户，tty表示网络中主机名，/表示当前目录（当登录用户，登录主机名以及进入目录不同时，相应的项也会改变），#表示登录用户是超级用户root，如果是一般用户则为\$。

1. 关机与重新启动

关机可以使用命令：`#init 0`；重新启动系统可以使用命令：`#init 6`。命令init用于立即关机或重启，但在多用户系统中，若想给用户发送关机警告信息以便各个用户完成自己的工作并注销登录，则必须使用shutdown命令。如：`#shutdown -h +5`，该命令警告所有用户5分钟后关闭系统，警告信息将显示在所有已经登录的终端上。用-r参数代替-h参数表示重新启动。

2. 常用文件目录操作命令

1) ls 显示文件和目录列表。ls命令应该是Unix用户和Linux用户非常熟悉的命令，该命令的基本功能是浏览文件。当然，它的后面可以附加一定格式的参数的，从而来显示关于文件的不同信息。比如，后面经常附加的参数有：`-a`，`-l`。其中这两个参数的作用如下：

`ls -a` 在Linux系统中，以.开头的文件被系统视为隐藏文件，仅用ls命令是无法看到的。如果想要显示出隐藏文件，我们就要使用`ls -a`命令。该命令除了显示一般文件名外，连隐藏文件也同时显示出来。

`ls -l` 以长格式显示结果。有的时候我们希望察看更详细的文件或目录属性，这个时候我们就可以使用-l参数。比如，在某个文件目录下使用该命令后可能显示如下信息：

```
drwx----- 2 Guest users 1024 Nov 21 21:05 Mail
-rwx--x--x 1 root root 89080 Nov 7 22:41 tar*
```

```
-rwxr-xr-x 1 root bin 5013 Aug 15 9:32 uname*  
lrwxrwxrwx 1 root root 4 Nov 24 19:30 zcat->gzip
```

上面显示的信息当中，每一行的信息可以分成7栏。其中，最左端第一栏中包含10个字符，比如“lrwxrwxrwx”。这10个字符主要是显示文档的类型以及不同用户对该文档的操作权限。比如，从左向右第一个字符表示文件是目录、连接文件或普通文档。其中，d表示目录文件，l表示连接文件，-表示一般文件。后面的9个字符可以被分成3组，每组3个字符，分别用于表示Owner,Group,Other用户对该文件的操作权限；其中，r：可读，w：可写，x：可执行，有时执行部分不是x而是s，表示这个程序的执行者临时拥有和拥有者一样的权利来执行该文件。第一栏后面的第二个栏是一个数字，用于表示文件数目，如果是目录，则是该目录下文件数目。第三栏用于表示文件或目录的拥有者，若使用者目前处于自己的Home，那这栏则是他的账号名称。第四栏表示所属的组。第五栏表示文件大小，一般是以字节为单位。第六栏表示文件的创建时间，第七栏则给出了文件名。

2) cp 复制文件或目录。

用法：cp [options] source dest或cp [options] source directory

选项：-a 表示尽可能将档案状态、权限等资料都按照原样予以复制。

-r 若source中含有目录名，则目录下档案也按序复制到目标目录中。

-f 若目的目录中已经包含与被复制文件或者目录相同名字的文件或者目录，则复制前先将原来的文件或者目录删除，然后把新的文件或者目录写入到该目录下。

范例：cp aaa bbb 表示将档案aaa复制并命名为bbb；cp *.c Finished表示将所有c语言程序拷贝到Finished目录下。

3) mv：移动文件或目录，文件或目录改名。

用法：mv [options] source target

选项：-b 在把文件名或子目录名改为其他文件或子目录已经使用过的名字时，将会对所有原有文件和子目录备份。

-i mv命令没有回显，使用该参数可以和用户交互。

范例：mv -b file1 file2该命令将file1改名为file2时，如果原来有file2文件，则对原file2文件进行备份（file~）。mv -i file1 file2 执行该命令会出现mv replace ‘file2’ ?y回车即可。

4) cat：显示文本文件内容。

用法：cat [options] filename

选项：-n 由1开始对所有输出的行数编号。

-b 和-n相似，只不过对于空白行不编号。

范例：cat -n file1 >file2 将file1文档内容加上行号后输出到file2文档中。cat -b file1 file2 >> file3 将file1，file2文档的内容加上行号后（空白行不编号）输出到file3中。

5) more, less: 分页显示文本文件内容。

范例: more filename或less filename。

6) head, tail: 显示文本文件的前若干行或后若干行。

范例: head -4 myfile, 显示文件myfile前4行内容; tail -4 myfile, 显示文件 myfile后4行内容; tail +45 myfile, 显示文件myfile从45行到文件尾的内容; tail +15 myfilehead -3, 显示文件myfile从15行开始的3行内容。

7) wc: 统计指定文本文件的行数、字数、字符数。

范例: wc -l myfile 统计myfile的行数。

8) find: 在文件系统中查找指定文件。

范例: find .-name 'my*', 从当前目录下开始查找以my开头的文件。find /home-user "lrj", 从/home目录下开始查找用户宿主为lrj的文件。

9) grep: 从指定的一个或多个文本文件中逐行查找指定的字符串。

范例: grep "my*" myfile1 myfile2从myfile1和myfile2中查找所有以my开头的字符串。

10) pwd: 显示当前工作目录。

11) mkdir: 创建目录。

范例: mkdir c_code, 在当前路径下创建了一个名为c_code的子目录。

12) rmdir: 删除空目录。

选项: -p删除某个子目录的全部继承结构。

范例: rmdir -p /tmp/parent/child必须指明某个子目录完整的结构才能删除其全部继承结构, 如果不带-p, 则只删除child目录。

13) rm: 删除文件。

用法: rm [options] 文件名

参数: -i 删除文件时会询问用户是否删除。

-f 强行删除某个文件。

范例: rm -f temp* 删除以temp开头的文件, 如果有同名的目录, 则rm删除时会报错, -f, -r参数一起使用时可以删除该同名目录以及其下的子目录。

3. 常用信息显示命令

1) stat: 显示指定文件的相关信息。

范例: stat file显示文件file的相关信息。

2) who: 显示在线登录用户。

3) whoami: 显示用户自己的身份。

4) hostname: 显示主机名称。

5) uname: 显示操作系统信息。

6) ifconfig: 显示网络接口信息。

范例: ifconfig显示全部网络接口信息。ifconfig eth0 显示网络接口eth0信息。