

## <<Linux/Unix设计思想>>

### 图书基本信息

书名：<<Linux/Unix设计思想>>

13位ISBN编号：9787115266927

10位ISBN编号：7115266921

出版时间：2012-3-28

出版时间：人民邮电

作者：甘卡兹

译者：漆犇

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<Linux/Unix设计思想>>

### 内容概要

作为开发unix多年的专家，mike

gancarz曾说过：“要想成为计算机的主人，而不是它的奴隶，你就应该使用linux！”

有别于市面上的其他关注如何使用linux的书籍，这《linux/unix设计思想》讲述的是“linux的思维方式”，揭示了linux正是unix这一无所不能的操作系统的完美实现。

到目前为止，没有一《linux/unix设计思想》同时介绍unix和linux的设计理念，《linux/unix设计思想》将这两者有效地结合起来，保留了the unix philosophy中unix方面的内容的同时，探讨了linux和开源领域的新思想。

## <<Linux/Unix设计思想>>

### 作者简介

Mike

Gancarz是美国佐治亚州亚特兰大市的一名编程及应用顾问。

他的团队使用Linux、Unix和Java工具，为金融服务行业开发出多个获奖的成像解决方案。

作为Unix应用程序设计专家，他不遗余力地推广Unix已达二十多年。

作为开发出X

Window System的团队成员，Mike

Gancarz还始创了一些至今仍应用在Linux的最新窗口管理器中的可用性观念。

Mike曾经在DEC公司位于新罕布什尔州纳舒厄城的Unix工程开发项目组工作，主持了将Unix命令和程序移植到64位Alpha处理器的工作。

他的首部著作The

Unix Philosophy ( Digital Press, 1995 ) 令数以万计的技术人员受益。

## &lt;&lt;Linux/Unix设计思想&gt;&gt;

## 书籍目录

## 第1章 unix哲学：集思广益的智慧

- 1.1 nih综合征
- 1.2 unix的开发
- 1.3 linux：一个人加上一百万人的智慧
- 1.4 unix哲学概述

## 第2章 人类的一小步

- 2.1 准则1：小即是美
- 2.2 简化软件工程
  - 2.2.1 小程序易于理解
  - 2.2.2 小程序易于维护
  - 2.2.3 小程序消耗的系统资源较少
  - 2.2.4 小程序容易与其他工具相结合
- 2.3 关于“昆虫”的研究
- 2.4 准则2：让每一个程序只做好一件事

## 第3章 快速建立原型的乐趣和好处

- 3.1 知识与学习曲线
  - 3.1.1 事实上，每个人有自己的学习曲线
  - 3.1.2 大师们也知道，变化不可避免
  - 3.1.3 为什么软件会被称为“软件”
- 3.2 准则3：尽快建立原型
  - 3.2.1 原型的建立是学习的过程
  - 3.2.2 建立早期原型能够降低风险
- 3.3 人类创造的三个系统
- 3.4 人类的“第一个系统”
  - 3.4.1 在背水一战的情况下，人类创建了“第一个系统”
  - 3.4.2 没有足够的时间将事情做好
  - 3.4.3 “第一个系统”是单枪匹马或是一小群人开发的
  - 3.4.4 “第一个系统”是一个“精简、其貌不扬的计算机器”
  - 3.4.5 “第一个系统”的概念可以激发他人的创造力
- 3.5 人类的“第二个系统”
  - 3.5.1 “专家”使用“第一个系统”验证过的想法来创建“第二个系统”
  - 3.5.2 “第二个系统”是由委员会设计的
  - 3.5.3 “第二个系统”臃肿而缓慢
  - 3.5.4 “第二个系统”被大张旗鼓地誉为伟大的成就
- 3.6 人类的“第三个系统”
  - 3.6.1 “第三个系统”由那些为“第二个系统”所累的人们创建
  - 3.6.2 “第三个系统”通常会改变“第二个系统”的名称
  - 3.6.3 最初的概念保持不变并显而易见
  - 3.6.4 “第三个系统”结合了“第一个系统”和“第二个系统”的最佳特性
  - 3.6.5 “第三个系统”的设计者有充裕的时间将任务做好
- 3.7 linux既是“第三个系统”，又是“第二个系统”
- 3.8 建立“第三个系统”

## 第4章 可移植性的优先权

- 4.1 准则4：舍高效率而取可移植性
  - 4.1.1 下一……的硬件将会跑得更快

## &lt;&lt;Linux/Unix设计思想&gt;&gt;

- 4.1.2 不要花太多时间去优化程序
  - 4.1.3 最高效的方法通常不可移植
  - 4.1.4 可移植的软件还减少了用户培训的需求
  - 4.1.5 好程序永不会消失，而会被移植到新平台
  - 4.2 准则5：采用纯文本文件来存储数据
    - 4.2.1 文本是通用的可转换格式
    - 4.2.2 文本文件易于阅读和编辑
    - 4.2.3 文本数据文件简化了unix文本工具的使用
    - 4.2.4 可移植性的提高克服了速度的不足
    - 4.2.5 速度欠佳的缺点会被明年的机器克服
  - 第5章 软件的杠杆效应
    - 5.1 准则6：充分利用软件的杠杆效应
      - 5.1.1 良好的程序员编写优秀代码，优秀的程序员借用优秀代码
      - 5.1.2 避免nih综合征
      - 5.1.3 允许他人使用你的代码来发挥软件杠杆效应
      - 5.1.4 将一切自动化
    - 5.2 准则7：使用shell脚本来提高杠杆效应和可移植性
      - 5.2.1 shell脚本可以带来无与伦比的杠杆效应
      - 5.2.2 shell脚本还可以充分发挥时间的杠杆效应
      - 5.2.3 shell脚本的可移植性比c程序更高
      - 5.2.4 抵制采用c语言来重写shell脚本的愿望
  - 第6章 交互式程序的高风险
    - 6.1 准则8：避免强制性的用户界面
      - 6.1.1 cui假定用户是人类
      - 6.1.2 cui命令解析器的规模庞大且难以编写
      - 6.1.3 cui偏好“大即是美”的做法
      - 6.1.4 拥有cui的程序难以与其他项目相结合
      - 6.1.5 cui没有良好的扩展性
      - 6.1.6 最重要的是，cui无法利用软件的杠杆效应
      - 6.1.7 “cui有什么关系？”
- 人们都不愿意打字了。
- ”
- 6.2 准则9：让每一个程序都成为过滤器
    - 6.2.1 自有计算技术以来，人们编写的每一个程序都是过滤器
    - 6.2.2 程序不创建数据，只有人类才会创建数据
    - 6.2.3 计算机将数据从一种形式转换成另一种
  - 6.3 linux环境：将程序用作过滤器
- 第7章 更多unix哲学：十条小准则
  - 7.1 允许用户定制环境
  - 7.2 尽量使操作系统内核小而轻量化
  - 7.3 使用小写字母并尽量简短
  - 7.4 保护树木
  - 7.5 沉默是金
  - 7.6 并行思考
  - 7.7 各部分之和大于整体
  - 7.8 寻求90%的解决方案
  - 7.9 更坏就是更好

## <<Linux/Unix设计思想>>

7.10 层次化思考

第8章 让unix只做好一件事

第9章 unix和其他操作系统的哲学

9.1 雅达利家用电脑：人体工程的艺术

9.2 ms-dos：七千多万用户的选择不会错

9.3 vms系统：unix的对立面

第10章 拨开层层迷雾：linux与windows的比较

10.1 内容为王，傻瓜

10.1.1 视觉内容：“用自己的眼睛去看。”

10.1.2 有声内容：“听得到吗？”

10.1.3 文字内容：“视频可以终结广播明星，却消灭不了小报。”

第11章 大教堂？

多怪异

第12章 unix的美丽新世界

## &lt;&lt;Linux/Unix设计思想&gt;&gt;

## 章节摘录

Unix哲学：集思广益的智慧这个世纪的哲学会成为下一个世纪的常识。

——中国幸运饼干 许多人都将发明Unix操作系统的殊荣授予AT&T公司的Ken Thompson，从某种意义上来说，他们是对的。

1969年在新泽西州美利山AT&T公司的贝尔实验室，Thompson编写出了Unix的第一个版本。

它作为Space Travel程序的平台运行在Digital PDP-7小型机上。

此前，Space Travel程序运行在由麻省理工学院开发的Multics系统上。

Unix的开发基于Multics系统，后者属于最早的一批分时操作系统。

在Multics开发之前，大多数计算机操作系统都运行在批处理模式下，这迫使程序员们要去编辑大堆的打孔卡片或纸带。

在那些日子里，编程是一个耗时费力的过程。

当时有一句流行语是：“上帝帮帮那些打翻了打孔卡片盒的傻瓜吧。

”干过卡片机编程的人都懂。

Thompson借鉴了Multics的许多特性，并将它们融入到早期的Unix版本，其中最主要的特点就是分时处理。

如果没有这种特性，那些在当前Unix系统或是其他操作系统上被人们视作理所当然的大部分功能，就会失去它们真正的力量。

Thompson的开发工作从借鉴Multics的想法入手，对于Unix开发人员而言，这样的套路可谓是驾轻就熟：良好的程序员写出优秀的软件，优秀的程序员“窃取”优秀的软件。

当然，我们并没有暗示Thompson是一个小偷。

但正是他这种在某些方面避免NIH（Not Invented Here，非我发明）综合症的意愿和基于别人的成果添加颇具创造性价值的做法，大力推动了这一款或许是历史上最精巧操作系统的出台。

我们还将在后面探讨“窃取”软件的意义。

现在只需要记住，将一个想法与人共享就如同一个大脑里有了两个想法。

1.1 NIH综合症软件开发人员经常会受到NIH综合症的影响。

在查看别人编写的软件解决方案时，他认为自己完全可以做得更好。

也许他真的能更为痛快利落地完成这项工作，但他并不知道别的开发人员当时面临的限制条件。

他们可能迫于时间或预算的压力，于是，只能集中精力处理这个解决方案中的某些特定部分。

## <<Linux/Unix设计思想>>

### 媒体关注与评论

Linux和GNU项目的理念表面上是Unix哲学的下一个发展阶段，实际上它只是生生不息的Unix的强势回归。

The Unix Philosophy第一版中阐述的准则至今仍确切无误，甚至得到更多的佐证。

开源除了可以让你清楚地了解到这些编程大师们创建系统的方式，还可以激励你去创建更快、更强大的系统。

——Jon “maddog” Hall Linux国际协会，执行理事 Gancarz有效地结合了Unix本身的准则和Linux开发社区中使用的Unix准则，对开源哲学进行了全新的阐释。

——Henry L. Hall



## <<Linux/Unix设计思想>>

### 编辑推荐

《Linux\Unix设计思想/图灵程序设计丛书》编辑推荐：剖析Linux/Unix制胜之道全新阐释开源哲学Jon "maddog" Hall作序并推荐！

<<Linux/Unix设计思想>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>