PEARSON
Addison Wesley

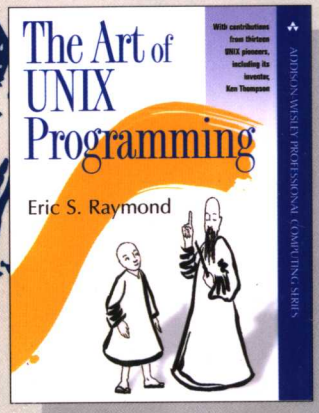
典藏原版书苑

# The Art of UNIX Programming

[美] Eric S. Raymond 著

# UNIX 编程艺术

(英文版)




人民邮电出版社
POSTS & TELECOM PRESS

“阅读《UNIX 编程艺术》弥补了我知识中的不足。它让我更加深入地理解了为什么 UNIX 是一种社区风格。在将程序看作实物上升到将程序看作社区投影的过程中，本书是一本非常适时的读物。从这个角度说，Eric 让 UNIX 的社区理念更加完美。”

——Kent Beck, *Extreme Programming Explained*、*Test Driven Development*和*Contributing to Eclipse*的作者

“一本令人愉悦使人着迷的读物，可以教授任何操作系统上的程序员以解决问题的经验。”

——Bruce Eckel, *Thinking in Java*和 *Thinking in C++*的作者

Eric S.Raymond 从 1982 年开始从事 UNIX 开发。作为开源社区文化的倡导者和呼吁者，他在《大教堂与市集》中发表了这场运动的宣言，同时他还编辑了《新黑客词典》一书。

## 编写优质软件：凝聚 30 年 UNIX 开发的智慧

本书的编写历时 5 年，作者将其 30 年中未见纸端的 UNIX 软件工程智慧结晶奉献给读者。作者第一次将软件哲学、设计模式、工具、文化和传统精华展示给读者，这些精华使 UNIX 成为具有创新意义的软件，并展示了它们如何影响着当今的Linux和开源运动。本书中包含的大量实例都来源于重要的开源项目，通过这些实例，可以教会 UNIX 和 Linux 程序员如何使软件更优雅、更可移植、更加长效以及更具可重用性。

• 装帧设计：胡平利

## 本书融入了 13 位 UNIX 先锋的经典评论：

- Ken Thompson UNIX 的创始人。
- Ken Arnold 4BSD UNIX发布组的成员，*The Java Programming Language*的合著者。
- Steven M.Bellovin Usenet的创始人之一，*Firewalls and Internet Security*的合著者。
- Stuart Feldman 贝尔实验室 UNIX 开发组成员，make 和 f77 的作者。
- Jim Gettys 和 Keith Packard X 窗口系统的主要架构者。
- Steve Johnson *yacc* 和 *Portable C Compiler* 的作者。
- Brian Kernighan *C Programming Language*、*The UNIX Programming Environment*、*The Practice of Programming* 和 awk 编程语言的合著者。
- David Korn korn shell的开发者，*The New Korn Shell Command and Programming Language* 的作者。
- Mike Lesk 贝尔实验室开发组成员，ms 宏包、tbl 和 refer 工具、lex 和 UUCP 的开发者。
- Doug McIlroy 贝尔实验室研究组的主管，UNIX 就诞生于该研究组，他还是 UNIX 管道的发明者。
- Marshall Kirk McKusick 4.2BSD fast filesystem的作者，4.3BSD和4.4BSD开发组的领导。
- Henry Spencer 早期 UNIX 开发者中的领袖，他创建了第一个开源的字符串处理库 getopt 以及用于 4.4BSD 的正则表达式引擎。



ISBN 7-115-14986-0

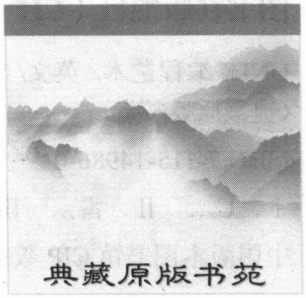
9 787115 149862 >

分类建议：计算机 / 操作系统 / UNIX

ISBN7-115-14986-0/TP·5546

定价：52.00 元

人民邮电出版社网址 www.ptpress.com.cn



典藏原版书苑

# UNIX 编程艺术

（英文版）

The Art of UNIX Programming

[美] Eric S. Raymond 著



人民邮电出版社

**图书在版编目（CIP）数据**

UNIX 编程艺术. 英文/（美）雷蒙德（Raymond, E. S）著. —北京：人民邮电出版社，2006.8
（典藏原版书苑）

ISBN 7-115-14986-0

Ⅰ. U… Ⅱ. 雷… Ⅲ. UNIX 操作系统—程序设计—英文 Ⅳ. TP316.81

中国版本图书馆 CIP 数据核字（2006）第 076823 号

**版 权 声 明**



典藏原版书苑

**UNIX 编程艺术（英文版）**

◆ 著 ［美］Eric S. Raymond

责任编辑 刘映欣

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@ptpress.com.cn

网址 http://www.ptpress.com.cn

北京顺义振华印刷厂印刷

新华书店总店北京发行所经销

◆ 开本：800×1000 1/16

印张：34.5

字数：729 千字 2006 年 8 月第 1 版

印数：1－3 000 册 2006 年 8 月北京第 1 次印刷

著作权合同登记号 图字：01-2006-3681 号

ISBN 7-115-14986-0/TP・5546

定价：52.00 元

**读者服务热线：(010)67132705 印装质量热线：(010)67129223**

# 内容提要

本书主要介绍了 Unix 系统领域中的设计与开发哲学、思想文化体系、原则与经验，总结了 Unix 发展史上成功的经验和失败的教训、经过时间验证的编码策略以及普遍适用的实用工具。本书由著名的 Unix 编程大师、开源运动领袖人物之一 Eric S. Raymond 历时多年编写而成，汇集了 Unix 之父 Ken Thompson 等 13 位 Unix 先锋的经典评论。本书内容涉及领域文化、软件开发设计与实现，覆盖面广，内容精湛，体现了作者深厚的经验积累和领域智慧，是 Unix 领域中一本不朽的经典名著。

**Trademark Notices**

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Addison-Wesley was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.



AIX, AS/400, DB/2, OS/2, System/360, MVS, VM/CMS, and IBM PC are trademarks of IBM. Alpha, DEC, VAX, HP-UX, PDP, TOPS-10, TOPS-20, VMS, and VT-100 are trademarks of Compaq. Amiga and AmigaOS are trademarks of Amiga, Inc. Apple, Macintosh, MacOS, Newton, OpenDoc, and OpenStep are trademarks of Apple Computers, Inc. ClearCase is a trademark of Rational Software, Inc. Ethernet is a trademark of 3COM, Inc. Excel, MS-DOS, Microsoft Windows and PowerPoint are trademarks of Microsoft, Inc. Java. J2EE, JavaScript, NeWS, and Solaris are trademarks of Sun Microsystems. SPARC is a trademark of SPARC international. Informix is a trademark of Informix software. Itanium is a trademark of Intel. Linux is a trademark of Linus Torvalds. Netscape is a trademark of AOL. PDF and PostScript are trademarks of Adobe, Inc. UNIX is a trademark of The Open Group.

The photograph of Ken and Dennis in Chapter 2 appears courtesy of Bell Labs/Lucent Technologies.

The epigraph on the Portability chapter is from the *Bell System Technical Journal*, v57 #6 part 2 (July–Aug. 1978) pp. 2021–2048 and is reproduced with the permission of Bell Labs/Lucent Technologies.

# 前　言

*Unix is not so much an operating system as an oral history.*

与其说 Unix 是操作系统，不如说它是一部口述的历史。

——Neal Stephenson

知识和技能有很大不同，知识可以让你去做正确的事，可以判断该做什么，而技能可以让你不假思索地完成正确的事。

这本书包含了许多与“知识”相关的内容，但更注重“技能”。你将从书中学到那些 Unix 专家们都不自知的 Unix 开发知识。与其他 Unix 图书相比，本书较少着力于技术，而是更多地涉足“共享文化”：外在的和内含的文化，自觉的和不自觉的传统。这不是一本讲“How-to”的书，而是一本讲“Why-to”的书。

了解“Why-to”有很实际的意义，有太多设计不良的软件：代码冗长，难于维护、移植和扩展——这些都是蹩脚设计的问题所在。希望本书的读者能够体会到 Unix 教给我们的优秀设计。

本书分为场景、设计、工具和社群 4 个部分。第 1 部分**场景**涉及哲学和历史，为后续内容埋下了伏笔。第 2 部分**设计**将 Unix 哲学原理细分为关于设计与实现更具体的建议。第 3 部分**工具**着眼于 Unix 所提供的工具，从而有助于解决问题。第 4 部分**社群**讲述人与人之间的事务与约定，这正是 Unix 文化拥有高效能的原因。

本书是关于共享文化的，因此我从未想过独自完成。书中出现了数位杰出的 Unix 开发人员的经典论述，他们也是 Unix 传统的缔造者。本书曾经被大范围地公开审阅，这期间我邀请这些资深人士对书稿进行讨论和审阅。这些意见没有被湮没，无论是为本书呐喊助阵还是摇头反对，你都可以在书中聆听到他们的真实声音。

本书中用到人称“我们”时，并不是虚张声势，而是反映这样一个事实：它力图说明这是整个社群的技能。

因为本书着力于传播文化，所以与其他技术书不同，本书加入了很多历史沿革和轶闻。希望你喜欢，这些东西其实是 Unix 程序员的素养。在历史发展的长河中，一个人的力量微不足道，一个群体才可以使历史更加鲜活生动，我们觉得这种方式能够使故事更加有趣。了解 Unix 的由来和变迁，对于培养对 Unix 风格的感觉大有裨益。

同样的原因，本书不会将历史沿革描绘成已经终结。你会发现本书包含大量当前的参考信息。我们并非想妄称现有的实践会反映出某些亘古不变的终极真理。参考当前的信息这一做法也提醒读者，现在你所掌握的信息，也许三五年后就会过时，需要重新审视。

另外，本书不是 C 教程，不是 Unix 命令和 API 的手册，不是 sed/yacc/Perl/Python 的语言参考，也不是网络编程入门，更不是专注于细节且令人费解的 X 指南。本书也不打算带领读者领略 Unix 内幕和体系结构。有很多其他的好书都涉及了这些领域，本书会在适当的时候告诉你应该参考哪些相关资料。

在这些技术细节之外，Unix 文化有一个未见诸笔端的技术传统，以熟练工来考量，它历经了数千万人年的开发和完善[1]。本书立足于这样的一个理念：领会此传统，并将它的设计手法应用于手边，你将会成为更好的程序员和设计师。

构成文化的是人，一直以来，获知 Unix 文化的方式通常是口口相传和潜移默化。本书不打算取代人与人之间的文化传播渠道，但可以促进这一过程，使你能够倾听他人的经验之谈。

## 本书的读者对象

如果你是 Unix 资深程序员，经常教导入门级程序员，或者与其他操作系统的拥趸进行辩论时无法阐明使用 Unix 方案所带来的好处时，可以阅读本书。

如果你是 C、C++或者 Java 程序员，有其他操作系统的开发经验，目前要开发一个 Unix 项目时，可以阅读本书。

如果你是初级或者中级水平的 Unix 用户，但是没有太多的开发经验，想学习在 Unix 下如何高效编程时，可以阅读本书。

如果你不在 Unix 下编程却发觉 Unix 的传统文化给你带来某种启迪，那么你就对了，Unix 哲学适用于其他操作系统。因此我们会用比其他 Unix 书籍更多的篇幅关注非 Unix 环境（特别是微软的操作系统）：当所用到的工具或者案例适用于其他操作系统时，我们会

[1] 从 1969 到 2003 年，35 年时间并不短。以这期间众多 Unix 站点数量的历史曲线来估算，人们在 Unix 系统的开发方面投入了约 5000 万人年。

让你知道。

如果你是系统架构师，正为通用市场或垂直应用准备平台方案或实现策略时，可以阅读本书。本书将帮助你了解 Unix 作为开发平台的强大功能，以及继 Unix 的开放源码传统之后的开发方式。

如果你想学习 C 编程的细节或者想知道怎么使用 Unix 内核 API，那么本书可能不适合你。我们推荐 *Advanced Programming in the Unix Environment* [Stevens92]，这是关于 Unix API 的经典名著，我们还推荐 *The Practice of Programming* [Kernighan-Pike99]，这是每个 C 程序员的必读书籍（使用任何编程语言的程序员都该阅读这本书）。

## 如何使用这本书

本书既注重实践，更富于理念；既包含警世格言，又不忘检点 Unix 开发中的特殊案例。在每个警句的前后，都有生动的实例阐明其由来，这些例子绝不是来自小儿科式的示例程序，而是均出自真实世界日常可见的运行代码。

我们尽量避免以大量代码或者规范文件来凑数，当然这么做会让本书的写作轻松许多（某些地方或许读起来也更轻松）。绝大多数编程书籍只授你以鱼，而本书则避免这种做法，力求培养读者“探求事情何以如此”的感知力。

因此，本书会在多处推荐阅读代码和规范文档，而这些代码和文档在书中只是少量地出现，其余部分我们会在举例时告诉你如何从网上获取。

从这些范例中汲取养分，有助于将所学原则消化吸收。如果你能边看着运行在 Unix 系统上的网页浏览器边阅读本书，那么再理想不过了。本书适用于任何 Unix 系统，但是我们将要研究的案例大多都会预装在 Linux 系统，或者可以从 Unix 系统上获得，书中会提示请你浏览或亲身感受它们。这些提示通常是按顺序的，离开一会儿并不会打散整个讲述过程的连续性。

注意：我们虽尽可能引用那些稳定可用的 URL，但我们没办法保证。如果发现书中出现的某个链接已不可用，那么你可以用搜索引擎进行短语搜索。我们会尽可能地在所引用的 URL 附近给出如何搜索的提示。

大多数缩略语在首次出现时会注明其全称。为了方便起见，附录中提供了术语表。

参考文献通常以作者名字出现。带编号的脚注是那些如果加在正文里可能会影响阅读的 URL，或者是我们认为是易变的 URL，也可能是旁白、战争故事甚至是笑话[1]。

[1] 这个特别的脚注献给 Terry Pratchett，他对脚注的用法简直是……绝了。

为了让技术水平不那么高的读者更容易读懂这本书，我们邀请了一些非程序员试读，并指出一些晦涩但起贯穿作用的术语。对于那些比较基本的术语，我们将其注释放在脚注中，因为那些有经验的程序员不太需要去读它。

## 相关参考文献

Unix 早期开发者的一些著名论文和书籍曾涉足这一领域，比如 Kernighan 和 Pike 的 *The Unix Programming Environment*[Kernighan-Pide84]就是其中的佼佼者，被视为经典。而现在看起来此书有点过时，它没有提到 Internet、万维网以及诸如 Perl、Tcl 和 Python 这些解释型语言新秀。

本书在写作过程中借鉴了 Mike Gancarz 的 *The Unix Philosophy*[Gancarz]。这本书在同类书籍中极其优秀，但是我们觉得需要更多内容才能反映出事情的全貌。尽管如此我们仍非常感谢这位作者，他使我们知道最简单的 Unix 设计方法就是最持久耐用的。

*The Pragmatic Programmer*[Hunt-Thomas]是一本关于良好设计的书，文风机智诙谐，与本书相比，它更倾向于软件设计工艺的另一个层面（更注重编码，而少着墨于高层面的问题划分）。作者的哲学是其在 Unix 领域耕耘的成果，也是对本书内容极好的补充。

*The Practice of Programming*[Kernighan-Pike99] 包含了一些与 *The Pragmatic Programmer* 共通的内容，但更进一步探究 Unix 传统的深邃。

最后，我们推荐 *Zen Flesh, Zen Bones*[Reps-Senzaki]，一部重要的佛教禅宗本源的合集。对禅的引用遍布全书，这样做是因为禅提供了一种概念，可以用来解决一些思路，这些思路对于软件设计似乎非常重要，并且其他方式很难将这种思路牢记在心。在这里，请不要把禅当成宗教，它是一种涤荡灵魂的东西，纯净而没有神灵的干扰——此即是禅。

## 本书的语法约定

从技术上和法律上讲，术语“UNIX”是 The Open Group 的商标，应该仅限于那些通过 The Open Group 严格的“符合标准”认证的操作系统。本书中使用其较宽松的定义，即大多数程序员所熟知的，Bell 实验室 Unix 代码的后裔或旁支。在这个意义上，Linux（书中的多数例子都基于 Linux）也算是一种 Unix。

本书也使用了 Unix 手册页（manual page）的传统，即以括号括起来的手册节号来标记 Unix 术语。通常用于强调一个 Unix 命令首次出现。比如 munger(1)可解读为“munger 程序被加入你的系统中，其文档位于 Unix 手册页的第 1 节”。第 2 节是 C 的系统调用，第

3 节是 C 的库函数调用，第 5 节是文件格式与协议，第 8 节是系统管理工具。其他节号本书未曾用到，其定义在各个 Unix 系统中各有不同。在你的 Unix 外壳提示符下可输入 man 1 man（老式的 System VUnix 系统可能要输入 man –s 1 man），以获得更多信息。

有时我们会提及某个 Unix 程序（比如 *Emacs*），后面没有手册节号，首字母大写。这意味着这个名字代表一族 Unix 程序，其基本功能相同，而我们将讨论其共性。比如 *Emacs*，就包含了 *xemacs*。

本书后面的很多地方采用了“old school”和“new school”的类比说法。在 rap 音乐中，所谓的“new-school”始于 1990 年前后，在时间上它恰好与脚本语言、图形用户界面、开放源码的 Unix 和万维网的兴趣相一致。“old-school”指代 1990 年以前（特别是 1985 年以前）的计算机界：昂贵（并且是共用）的计算机、专用的 Unix、shell 脚本和无所不在的 C。这些差异有必要指出，因为现在机器越来越便宜，内存越来越大，这些都极大地影响了 Unix 编程的风格。

## 所用案例

很多编程类书籍为证明某一观点而特地造出一个范例，本书却不是这样。本书的案例研究均来自真实世界，在生产环境中一直正常运行。下面是书中的一些主要案例。

### cdrtools/xcdroast

这两个独立的项目通常被一起使用。cdrtools 是一套刻盘工具（用关键字“cdrtools”可以在网上找到）。xcdroast 是 cdroast 的图型界面前端，其项目的网站为 http://www.xcdroast.org/。

### fetchmail

fetchmail 用于从远程邮件服务器上收信，它支持 POP3 和 IMAP 邮箱协议。其主页为 http://www.catb.org/~esr/fetchmail，也可以用关键字“fetchmail”在网上搜索到。

### GIMP

GIMP（GNU Image Manipulation Program，GNU 图像处理程序）是一个全特性的绘画和图像处理程序，可对多种图像格式进行复杂处理。其源码可从 GIMP 主页 http://www.gimp.org/获得（也可以通过关键字“GIMP”在网上搜索到）。

### *mutt*

mutt 邮件客户端是目前各类基于文本的邮件客户端程序中的翘楚，它提供对 MIME

（Multipurpose Internet Mail Extensions）以及个人隐私辅助程序，如 PGP（Pretty Good Privacy）和 GPG（GNU Privacy Guard）等特性的绝佳支持。其源码和二进制可执行文件可以从 Mutt 项目主页 http://www.mutt.org 上获得。

### xmlto

xmlto 可将 DocBook 和其他 XML 文档以多种格式渲染输出，包括 HTML、纯文本和 PostScript。其源代码和文档可在 xmlto 主页 http://cyberelk.net/tim/xmlto/上获得。

为了便于读者理解，我们将本书例子所需阅读的代码量降到最小，并尽量挑选那些可重复使用、并能体现多种不同设计原理和设计实践的实例。基于同样原因，很多实例来自我本人的项目。我不敢说这些例子最为恰当，只是觉得它们对阐述我的观点非常有用。

## 作者致谢

各位客串贡献者（Ken Arnold, Steven M.Bellovin, Stuart Feldman, Jim Gettys, Steve Johnson, Brian Kernighan, David Korn, Mike Lesk, Doug McIlroy, Marshall Kirk McKusick, Keith Packard, Henry Spencer 和 Ken Thompson）为本书增添了极高的价值。特别是 Doug McIlroy，在给予本书深邃精辟的评注的同时，也展现了他在 30 年前管理最原始的 Unix 研究组时的敬业精神。

特别感谢 Rob Landley 和我的妻子 Catherine Raymond，他们不厌其烦地对本书的初稿逐行审阅。在 Rob 深富洞察力的细致评述的启迪下，我在最终稿中加入了一整章内容，他在本书的组织结构和取材范围方面贡献相当多。如果把他所给予的改进意见落于笔端，那他无愧于本书的合著者。Cathy 代表读者中非技术人员群体，如果那些非程序员读者觉得本书并不难读，那么全是她的功劳。

在本书写作的 5 年间，我从不少人的讨论意见中获益良多。Mark M.Miller 使我对线程有了更深的认识。John Cowan 教会我很多接口设计方式，并起草了 wily 和 VM/CMS 的学习案例。Jef Raskin 告诉我 Rule of Least Surprise 的由来。UIUC System Architecture Group 对前几章给出的反馈意见弥足珍贵，What Unix Gets Wrong 和 Flexibility in Depth 两节就是他们直接激励的结果。Russell J.Nelson 提供了 Bernstein chaining 的素材。第 3 章中的 MVS 学习案例大部分的材料来自 Jay Maynard。Les Hatton 对语言一章给出很多有益建议，并促使我写成第 4 章中 Optimal Module Size 的部分内容。David A.Wheeler 贡献了很多发人深省的批评意见，以及一些学习案例（特别是在设计部分中）的素材。Russ Cox 帮助我进行了 Plan 9 的检查。Dennis Ritchie 纠正了我一些错误的 C 历史观念。

成百上千的 Unix 程序员在 2003 年 1 月到 6 月间的公开审阅过程中给了我很多建议和

评论，在此就不一一列出他们的名字。开放的同级审阅这一过程让我觉得紧张刺激而且回报多多。当然，最终书稿中残存的任何错误都是我自己的责任。

细致详尽的讲解风格以及书中一些其他概念，是受到了“设计模式运动”的影响。说实话，我对到处堆砌 Unix 设计模式这种做法非常不以为然。我对这一运动的中心教条不敢苟同，并且没觉得把设计模式严格付诸实用有很大的必要，也不想背上这种思想的包袱。尽管如此，我的行事方法仍然受到 Christopher Alexander 成果[1]（特别是 *Timeless Way of Building* 和 *A Pattern Language* 两文）的影响。Gang of Four 和他们的信徒为我展示了如何用 Alexander 的思想，站在较高的角度上，抛去对设计规则含混不清的空话来谈论软件设计，这一点我心存感激，永世不忘。对设计模式有兴趣的读者可以看看这本书 *Design Patterns: Elements of Reusable Object-Oriented Software* [GangOfFour]。

本书的书名毫无疑问是借鉴了 Donald Knuth 的 *The Art of Computer Programming* 的书名。Knuth 和 Unix 传统文化没什么联系，但他影响了我们每一个人。

Mark Taub 是一位富有洞察力和想象力的编辑，他从并不看好的项目中发现了闪光点，并极富技巧地促成了这本书的写作。在文字编辑中，文笔好而且能帮助别人提高写作能力的非常少，所幸 Mary Lou Nohr 是其中之一。Jerry Votta 的封面设计充分表达了我的意图，而且做得比我想象中的还要漂亮。Addison-Wesley 的编辑们让审稿和出版这一过程不至于非常枯燥无味，我天生就怕被人管，但是他们仍然极力配合我，使得文字、版面、图片和市场推广工作都达到了极高的水准，在此一并表示感谢。

[1] 一篇对 Alexander 工作成果的肯定文章，包含其部分重要成果的在线链接，可以参考 Some Notes on Christopher Alexander(http://www.math.utsa.edu/sphere/salingar/Chris.text.html)一文。

*To Ken Thompson and Dennis Ritchie,*
*because you inspired me.*

# Contents