

DPI

*De* Programmatica *Ipsium*

DE PROGRAMMATICA IPSUM

---

# Issue 092: Linux

May 4th, 2026

# Table of Contents

Issue 092: Linux .....	5
Have You HURD The GNUs? .....	9
Overchoice In The New Non-Aligned World .....	15
Hannu Puttonen .....	25
Linus Torvalds & David Diamond .....	31
Michael Kerrisk .....	35



# Issue 092: Linux



May 4th, 2026

Welcome to the 92nd issue of *De Programmatica Ipsum*, about *Linux*.

In this edition:

- Graham reports on the state of GNU Hurd with very good news<sup>1</sup>.
- Adrian untangles the Linux distro jungle<sup>2</sup> and helps you choose the best one for your needs.
- In our Vidéothèque section<sup>3</sup>, we watch “The Code” by Hannu Puttonen<sup>4</sup>.
- In the Library section<sup>5</sup>, we review “Just for Fun” by Linus Torvalds & David Diamond<sup>6</sup>, and “The Linux Programming Interface” by Michael Kerrisk<sup>7</sup>.

Download this issue in DRM-free PDF<sup>8</sup> or EPUB<sup>9</sup> format, and read it on your preferred device. You can also subscribe to our RSS feed<sup>10</sup>, featuring the full content of our articles.

We would like to thank our patrons who generously contribute every month (or have contributed in the past) to our work and help us run this magazine. Thank you so much! In alphabetical order: Adam Guest, Adrian Tineo Cabello, Benjamin Sheldon, Christopher Nascone, Colin Powell, Franz Lucien Moersdorf, Guillermo Ramos Álvarez, Jean-Paul de Vooght, Dr. Juande Santander-Vela, Patryk Matuszewski, Paul Hudson, Quico Moya, Roger Turner, Szymon Licau, and countless more leaving anonymous tips every month.

Enjoy this issue! Please share our articles on social media, or contribute<sup>11</sup> if you would like to support our work with a donation via Liberapay<sup>12</sup>.

Cover photo by Ian Parker<sup>13</sup> on Unsplash<sup>14</sup>.

## REFERENCES

- <sup>1</sup> <https://deprogrammaticaipsum.com/have-you-hurd-the-gnus/>
- <sup>2</sup> <https://deprogrammaticaipsum.com/overchoice-in-the-new-non-aligned-world/>
- <sup>3</sup> <https://deprogrammaticaipsum.com/category/videotheque/>
- <sup>4</sup> <https://deprogrammaticaipsum.com/hannu-puttonen/>
- <sup>5</sup> <https://deprogrammaticaipsum.com/category/library/>
- <sup>6</sup> <https://deprogrammaticaipsum.com/linux-torvalds-david-diamond/>
- <sup>7</sup> <https://deprogrammaticaipsum.com/michael-kerrisk/>
- <sup>8</sup> <https://deprogrammaticaipsum.com/pdf/issue-092-linux.pdf>
- <sup>9</sup> <https://deprogrammaticaipsum.com/epub/issue-092-linux.epub>
- <sup>10</sup> <https://deprogrammaticaipsum.com/index.xml>
- <sup>11</sup> <https://deprogrammaticaipsum.com/contribute/>
- <sup>12</sup> <https://liberapay.com/akosma/donate>
- <sup>13</sup> [https://unsplash.com/@evanescentlight?utm\\_source=unsplash&utm\\_medium=referral&utm\\_content=creditCopyText](https://unsplash.com/@evanescentlight?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)
- <sup>14</sup> [https://unsplash.com/photos/standing-penguin-on-sand-near-snow-covered-mountain-covering-the-sun-from-view-at-daytime-TLcLDigmTKE?utm\\_source=unsplash&utm\\_medium=referral&utm\\_content=creditCopyText](https://unsplash.com/photos/standing-penguin-on-sand-near-snow-covered-mountain-covering-the-sun-from-view-at-daytime-TLcLDigmTKE?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)



# Have You HURD The GNUs?



By Graham Lee

Those of us who have been extremely online for a very long time will remember that Linus Torvalds announced<sup>1</sup> his Linux kernel to a usenet newsgroup (or “froup”, as the internet lexicon had it at the time) for the Minix operating system<sup>2</sup>. Minix is a Unix-like operating system that Andrew S. Tanenbaum created to teach his class on operating systems, and that he published in the book “Operating Systems: Design & Implementation”<sup>3</sup>.

Minix uses a microkernel design, in which the kernel is responsible for a limited subset of system services—typically, process scheduling and inter-process communication (IPC) are all that is needed—and everything else runs as a process in “user space”, including device drivers, file systems, and virtual memory managers, in addition to the “applications” that you actually bought the computer to use. The typical alternative is the approach that Torvalds chose for Linux: a monolithic kernel, in which operating system services all run in the kernel, and user space processes provide applications.

Each design has its trade-offs. The kernel is a single, high-privilege space where any code running has very little restriction, so the microkernel approach reduces the “blast radius” of the security risk by strongly restricting the amount of code that runs in the kernel. Because all operating system services happen using IPC, a microkernel can further define the security controls in place in a running system by limiting which processes can communicate with which other processes—and it can do so dynamically, so that a process might be able to make some calls but not others, or make a call at one time that it cannot at another time.

However, all of this IPC comes at a cost, as any system call that an application makes might translate into multiple IPC communications, in which the calling process needs to rewrite its data to fit the interface format of the protocol with the receiving process, then tell the kernel to receive the message, which write the data into the receiving process’s memory space and schedules that process to run. Additionally, microkernel designs run the risk of an “upcall deadlock”<sup>4</sup>, where the kernel cannot make progress because it is waiting on a user-space process that is blocked in a system call. As an example, imagine that an application needs to open a file, and the filesystem process encounters an error that it tries to write to the kernel’s log—which is on the filesystem that encounters the error.

Conversely, the monolithic kernel design makes it easier for any kernel component to directly access any other kernel component: faster, harder to deadlock, but easier to create bugs in and easier to exploit. All those drivers running in the privileged kernel memory space, potentially from multiple vendors or other sources, introduce risks of potentially exploitable problems at the core of the operating system.

Tanenbaum and Torvalds famously had some heated discussions about the relative merits of each architecture<sup>5</sup>, which probably seemed important at the time but are less relevant these days as the microkernel design eventually won out in many contexts (consider mobile phone baseband environments and car management systems, which typically run microkernels like QNX or an L4 derivative), with a hybrid approach (Windows NT is a microkernel design that runs a lot of services in the kernel for performance, and XNU—macOS, iOS etc.—is a microkernel with a monolithic kernel grafted on, but that then runs a lot of secure services in out-of-kernel “exclave” environments) also being popular. At this point, the outlier is the one frequently-observed monolithic kernel design in the wild: Linux.

Anyway, back to that announcement. Torvalds explained that Linux was a hobby project, and “won’t be big and professional like GNU”. Well, these days, Linux is at the core of most operating environments that use GNU software, to the point that they are called “Linux distros” no matter how much some people might like everyone to call them “GNU plus Linux”<sup>6</sup>. So what gives?

The GNU project started in 1984<sup>7</sup>—a full seven years before that Usenet announcement—as a small collection of tools under an umbrella project whose creator’s aim was to offer a full replacement for a UNIX operating system that preserves and promotes the “four freedoms” of free software. Initially, those tools were the GNU Emacs text editor and the GNU C Compiler (later GNU Compiler Collection), both created by project leader Richard M Stallman. During the next few years, people contributed other components to the GNU system and built them in a way that meant they work on many different flavors of UNIX and UNIX-like operating systems, as well as less UNIX-like environments including the Amiga and Windows. However, the whole environment could not be free because there was no GNU kernel (the closest alternative being the BSD kernel, which had licensing restrictions that were interpreted as not freedom-respecting, and which was soon to become embroiled in a lawsuit with the original UNIX creators<sup>8</sup>, AT&T).

The effort to create a kernel, known as Trix<sup>9</sup>, was not close to ready. Eventually the GNU project entered into an agreement with Carnegie-Mellon University to use their Mach kernel (actually in May 1991, after Torvalds had begun work on

Linux). Mach is found in operating systems you likely have not used including OSF/1 and NEXTSTEP, and operating systems you may well have used if you have been near the Apple galaxy since about 1997—or earlier, if you are one of the few people who used Tenon’s MachTen<sup>10</sup> or Apple’s MkLinux<sup>11</sup>. Reigniting that debate about monoliths or microkernels, GNU Mach—the GNU project’s own implementation of Mach 3, begun in 1997—operates as a microkernel that loads the Hurd<sup>12</sup> (HIRD of UNIX-Replacing Daemons, where HIRD stands for HURD of Interfaces Representing Depth, where HURD...you get the idea). Immediately snuffing the debate back out, GNU plus Linux is used and useful, where GNU (including the Hurd) has been perpetually “nearly ready” for the intervening decades, including several attempts to replatform the whole system on to a different microkernel (the L4 family of microkernels, Coyotos’ microkernel, and a new project called Viengoos have all been candidates<sup>13</sup>).

Well, this is a nice story, but I am afraid it is out of date. At some point last year, GNU Hurd became relevant again, which long-time contributor Samuel Thibault explained to a packed room at the FOSDEM conference<sup>14</sup> earlier this year. A port of the Rust compiler greatly increased the amount of application software available. A 64-bit port of GNU Mach and GNU MIG (the Mach Interface Generator, the interface description language that describes the data structures Mach processes use for IPC) is complete, so the OS can take advantage of modern processors and large amounts of system memory. When I say “modern processors”, x86\_64 support is mostly ready, and AArch64 (64-bit ARM) is experimental.

By running a “rump” NetBSD kernel in user-space<sup>15</sup>, Mach gets modern networking, USB, and other drivers so that it can actually run on computers that contain those modern processors. Long-dormant multiprocessor code (CMU Mach was originally designed for supercomputers, which were envisioned at the time as symmetric multiprocessing systems) has been rewritten, so that GNU Mach is now multicore, albeit with most Hurd processes still residing on a single core while the team audits their parallelism safety. Journaling filesystems are nearly ready.

So complete is GNU Mach and GNU Hurd, that the GNU Guix system<sup>16</sup> offers it as an alternative kernel choice in the installer, if you do not want to use the Linux-libre default kernel. There has been a Debian GNU/Hurd distribution<sup>17</sup> for

donkey's years. It may have been April Fool's when Gentoo announced that they were switching to the Hurd<sup>18</sup>, but the availability of scripts and a virtual machine image with running Gentoo/Hurd was no joke. Your author is running it now, with this repository checked out in git and the GNU emacs editor making some final corrections to this article.

Today, GNU is fully usable, it is just not big and professional like GNU plus Linux. Some of the ideas that had to be bolted on to Linux to make it “big and professional” are native parts of the design when you use GNU Mach and the Hurd. Making isolated environments like the podman<sup>19</sup> containers we use to create and publish “De Programmatica Ipsum” requires an unholy amalgam of cgroups, namespaces, and other features that Linux has sprouted over time. In a Hurd system, you just define and launch a subhurd<sup>20</sup>: an independent set of the UNIX-replacing daemons that has limited access to the host HURD's capabilities. GNU Guix also offers the intriguing possibility of the “childhurd”<sup>21</sup>—a Hurd environment you define and run in your Linux-based Guix environment.

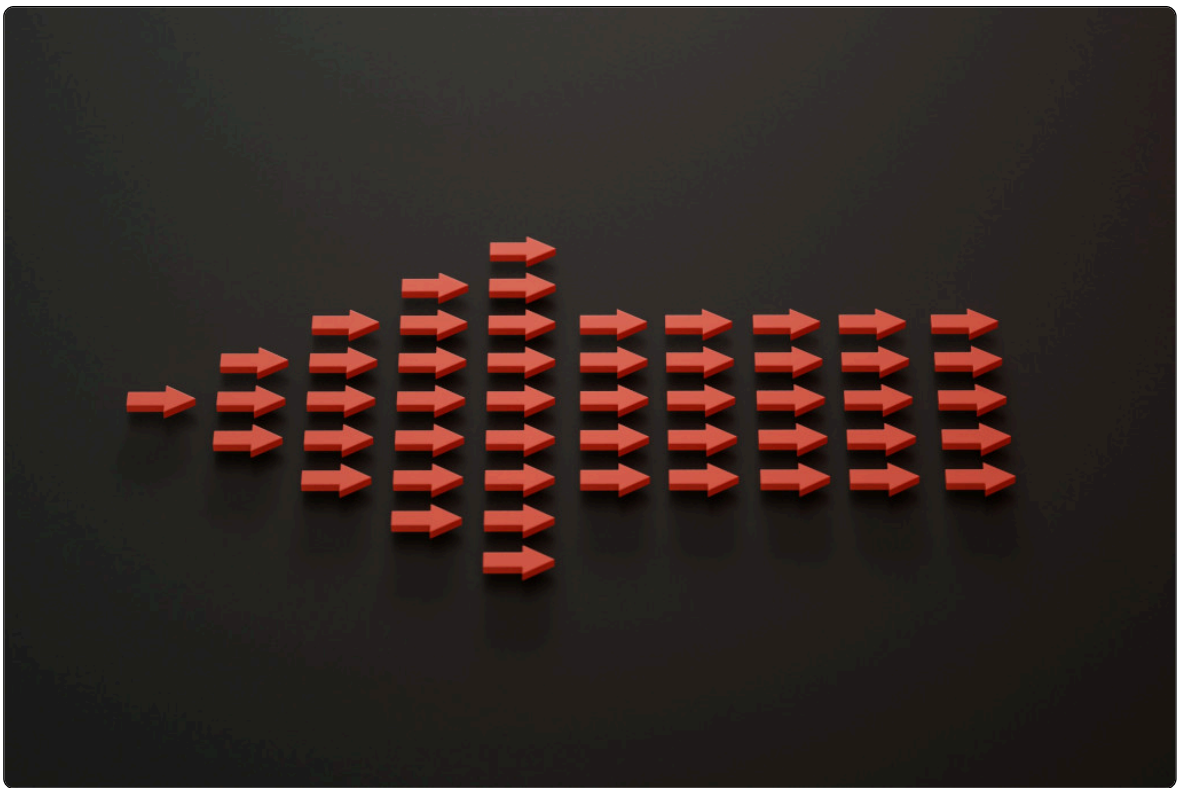
If your idea of the Hurd is that it was a nice idea once upon a time that never came to anything, I would recommend revisiting it now. You will be pleasantly surprised.

Cover photo by Helena Pfisterer<sup>22</sup> on Unsplash<sup>23</sup>.

REFERENCES

- <sup>1</sup> <https://www.cs.cmu.edu/~awb/linux.history.html>
- <sup>2</sup> <https://www.minix3.org>
- <sup>3</sup> <https://www.minix3.org/doc/>
- <sup>4</sup> <https://nvd.nist.gov/vuln/detail/CVE-2026-22980>
- <sup>5</sup> <https://www.oreilly.com/openbook/opensources/book/appa.html>
- <sup>6</sup> <https://www.gnu.org/gnu/incorrect-quotation.html>
- <sup>7</sup> <https://www.gnu.org/gnu/initial-announcement.html>
- <sup>8</sup> [https://en.wikipedia.org/wiki/UNIX\\_System\\_Laboratories,\\_Inc.\\_v.\\_Berkeley\\_Software\\_Design,\\_Inc.](https://en.wikipedia.org/wiki/UNIX_System_Laboratories,_Inc._v._Berkeley_Software_Design,_Inc.)
- <sup>9</sup> <https://landley.net/history/mirror/gnu/GNU-HURD-2.html>
- <sup>10</sup> <http://www.tenon.com/products/machten/>
- <sup>11</sup> <https://www.mklinux.org>
- <sup>12</sup> <https://www.gnu.org/software/hurd/>
- <sup>13</sup> [https://www.gnu.org/software/hurd/history/port\\_to\\_another\\_microkernel.html](https://www.gnu.org/software/hurd/history/port_to_another_microkernel.html)
- <sup>14</sup> [https://mirror.as35701.net/video.fosdem.org/2026/k4201/7FZXHF-updates\\_on\\_gnuhurd\\_progress\\_rump\\_drivers\\_64bit\\_smp\\_software\\_bootstrapping.mp4](https://mirror.as35701.net/video.fosdem.org/2026/k4201/7FZXHF-updates_on_gnuhurd_progress_rump_drivers_64bit_smp_software_bootstrapping.mp4)
- <sup>15</sup> [https://archive.fosdem.org/2022/schedule/event/dzammit/attachments/slides/4850/export/events/attachments/dzammit/slides/4850/rump\\_hurd\\_talk.pdf](https://archive.fosdem.org/2022/schedule/event/dzammit/attachments/slides/4850/export/events/attachments/dzammit/slides/4850/rump_hurd_talk.pdf)
- <sup>16</sup> <https://guix.gnu.org>
- <sup>17</sup> <https://www.debian.org/ports/hurd/>
- <sup>18</sup> <https://www.gentoo.org/news/2026/04/01/gentoo-hurd.html>
- <sup>19</sup> <https://podman.io>
- <sup>20</sup> <https://www.gnu.org/software/hurd/hurd/subhurd.html>
- <sup>21</sup> [https://guix.gnu.org/manual/devel/en/html\\_node/Virtualization-Services.html#Childhurd](https://guix.gnu.org/manual/devel/en/html_node/Virtualization-Services.html#Childhurd)
- <sup>22</sup> [https://unsplash.com/@pixbyhelena?utm\\_source=unsplash&utm\\_medium=referral&utm\\_content=creditCopyText](https://unsplash.com/@pixbyhelena?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)
- <sup>23</sup> [https://unsplash.com/photos/a-herd-of-wildebeest-running-across-a-dry-grass-field-xSbyDPqSZ-c?utm\\_source=unsplash&utm\\_medium=referral&utm\\_content=creditCopyText](https://unsplash.com/photos/a-herd-of-wildebeest-running-across-a-dry-grass-field-xSbyDPqSZ-c?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)

# Overchoice In The New Non-Aligned World



By Adrian Kosmaczewski

The Non-Aligned Movement<sup>1</sup> was born in 1961, during the most dramatic period of the Cold War, in opposition to the nuclear escalation threats between NATO and the Warsaw Pact. Countries from what it is now known as the “Global South” got together to figure out ways to help each other in a period of great turmoil.

Similar to those days of yore, “dramatic” is quite an appropriate word to describe the times we are living in; we are very far away from “the end of history” thesis proposed<sup>2</sup> by Francis Fukuyama between 1989 and 1992. In our nerd minds, the

sense of dread is accelerated by the fact that no dystopian sci-fi author ever came up with a scenario like the one we are doomscrolling on our devices every morning. The future used to look like flying cars and teleportation to Mars; instead, we had a fascist comeback streamed live over TikTok.

It is in such historical context that, while we were busy preparing this very text, the French government announced plans<sup>3</sup> to migrate an as-of-yet unknown number of computer desktops from Windows to Linux. Big tech, an admittedly US-based contraption, has turned against old allies and made Google's old "don't be evil" mantra a forgotten memory from a bygone era.

Linux becomes, as a matter of fact, the "non-aligned" option of operating systems, in stark opposition to the cold war represented by the duopoly of Windows and (admittedly, to a lesser extent) macOS. (We have already talked about the mobile duopoly in a previous article<sup>4</sup>, a market in which Linux is present in a different yet very strong way.)

The French decision, albeit late, is wise, commendable, and reasonable. If there is one thing we should learn for the current times, is that we cannot trust anymore in the desires and values of a psychotic government. And this includes operating systems, software, SaaS offerings, or whatever comes from the other side of the Atlantic.

(As a Swiss citizen, I would love to see my government follow the same path as France in this respect, but Switzerland has a track record of siding with the wrong side of history in order to maintain their beloved neutrality, so I am not holding my breath at this point.)

The problem is that, as far as options go, Linux is a terrible third one.

Do not get me wrong; I am a very happy Fedora Linux user, and I am using it at this very moment to write the lines you are reading. But I work in IT, and I am willing to put up with the myriad shenanigans Linux brings upon users. The work that French IT administrators is about to start is pharaonic, and I am not mincing my words.

Why do I say that Linux is a terrible option? Because of overchoice<sup>5</sup>, and because we are all very tired indeed.

Overchoice is a psychological factor with quite a history. First explained by Alvin Toffler<sup>6</sup> in his 1970 book “Future Shock”<sup>7</sup>, and then popularized as “The Paradox of Choice” by the 2004 book of the same name<sup>8</sup>, it is an intuitive part of our current world.

Here is a fun observation; one of the biggest criticisms that communism has received across the ages is that of lack of choice. Common was the mockery between the spartan looks of a Soviet grocery store, offering a very limited set of products (if any), with the overabundance of American supermarkets, created thanks to the forces of the “free market” that should never be stopped.

Yet, in the computing world, one of the most striking byproducts of capitalism, we have at most two versions of Windows (“Pro” and “Home”) and just one of macOS (remember when there was a “Server” version thereof?)

Readers well-versed in the history of Apple will surely remember Steve Jobs’ matrix of products when he returned to save the company from bankruptcy in 1997, drastically reducing the amount of products offered by the company to just four: two laptops and two desktops, one of each for “home” and “pro” users. Somehow, solving the paradox of choice was exactly what Apple needed to become the most valuable company on Earth merely 20 years later.

On the non-aligned side of things, and against all market logic, the Linux galaxy, itself, is a realm with a myriad choices. Maybe too much?

First of all, let us bring some terminology first. Nobody installs just “Linux” *per se*. What users install on their laptops are “Linux distributions”, also known as “distros” in the jargon and slang of this fringe realm. Longtime Linux expert Michael Kerrisk<sup>10</sup> explains distributions much better than I could:

*Precisely speaking, the term Linux refers just to the kernel developed by Linus Torvalds and others. However, the term Linux is commonly used to mean the kernel, plus a wide range of other software (tools and libraries) that together make a complete operating system. In the very early days of Linux, the user was*

*required to assemble all of this software, create a file system, and correctly place and configure all of the software on that file system. This demanded considerable time and expertise. As a result, a market opened for Linux distributors, who created packages (distributions) to automate most of the installation process, creating a file system and installing the kernel and other required software.*

*The earliest distributions appeared in 1992, and included MCC Interim Linux (Manchester Computing Centre, UK), TAMU (Texas A&M University), and SLS (SoftLanding Linux System). The oldest surviving commercial distribution, Slackware, appeared in 1993. The noncommercial Debian distribution appeared at around the same time, and SUSE and Red Hat soon followed. The currently very popular Ubuntu distribution first appeared in 2004. Nowadays, many distribution companies also employ programmers who actively contribute to existing free software projects or initiate new projects.*

So a “Linux distribution” is just a wonderful thing composed of a myriad different software thingies, forming what most users would end up calling an “operating system”. And just how many different thingies could be distributed this way? Hang on tight little tomato, this is going to be quite a ride.

- Some distributions are “free as in beer and free as in freedom”, in which case support means talking to fanatic users on a Discord forum telling each other to RTFM. On the other hand, some other distros actually cost money, in exchange for actual support, maybe even a 0-800 phone number to call when in despair, better compatibility with a variety of hardware, or some other warm feeling for you to feel inside. This latter case is usually preferred by businesses, for reasons that should be obvious by now.
- Some distributions are explicitly designed for beginners, others for more advanced users (the definition of which is always vague), or for specific use cases like conducting business, playing games, writing software, *und so weiter*.
- Some distros have specific hardware targets: some work fine on a standard laptop, while others work better on dedicated workstations, or maybe on servers, or even supercomputers, mainframes, or some embedded system NASA ships inside a probe to Mars.

- Some are easier to install than others: some might include a graphical “wizard” interface, while others... well, I hope you enjoy typing on your keyboard.
- Some are already localized for specific countries, which is something some users might prefer (or be forced to use). Most notably come to mind the ones from North Korea<sup>11</sup>, China<sup>12</sup>, Sweden<sup>13</sup>, Turkey<sup>14</sup>, Lithuania<sup>15</sup>, Romania<sup>16</sup>, Greece<sup>17</sup>, Philippines<sup>18</sup>, Colombia<sup>19</sup>, New Zealand<sup>20</sup>, Italy<sup>21</sup>, Brazil<sup>22</sup>, and of course Argentina<sup>23</sup>.
- Then there are the CPU architectures supported<sup>24</sup>, for an outstanding variety of computers out there... even though if most humans are running on some “86”-compatible thing. Of such variety, each distribution picks<sup>25</sup> their own preferred targets, changing their mind as years go by.

And then come the esoteric parameters that make or destroy distributions:

- The Kernel (the what?): the official from kernel.org, or the one from Linux-libre, Illumos, GNU Hurd, Haiku...
- Rolling release or not?
- Able to run Windows<sup>26</sup> and/or Mac<sup>27</sup> apps, or not?
- “Non-free” packages and drivers, available or not?
- File system: ZFS, FAT, ext, Btrfs, XFS...?
- Init system: runit, systemd, sysvinit, OpenRC...?
- Window system & protocol: X or Wayland? (Whatever those things are, they will restrict your ability to run some of your preferred software. You have been warned.)
- Package manager: APT<sup>28</sup>, RPM<sup>29</sup>, apk, pacman, pacstall<sup>30</sup>, Homebrew<sup>31</sup>, flatpak, snap...
- C standard library: glibc<sup>32</sup>, musl<sup>33</sup>, µClibc<sup>34</sup>, µClibc-ng<sup>35</sup>, klibc<sup>36</sup>, Newlib<sup>37</sup>, EGLIBC<sup>38</sup>, diet libc<sup>39</sup>, Bionic<sup>40</sup> ...
- Desktop environment (inhale deeply): GNOME<sup>41</sup>, KDE<sup>42</sup>, LXDE<sup>43</sup>, Xfce<sup>44</sup>, CDE<sup>45</sup>, herbstluftwm<sup>46</sup>, i3<sup>47</sup>, bspwm<sup>48</sup>, Enlightenment<sup>49</sup>, Sway<sup>50</sup>, Cinnamon<sup>51</sup>, Fluxbox<sup>52</sup>, MATE<sup>53</sup>, JWM<sup>54</sup>, Qtile<sup>55</sup>, Openbox<sup>56</sup>, Blackbox<sup>57</sup>, Claude’s Tab Window Manager<sup>58</sup>, EMWM<sup>59</sup>, NsCDE<sup>60</sup>, FVWM<sup>61</sup>, COSMIC<sup>62</sup>, olwm<sup>63</sup>, OpenWindows<sup>64</sup>, Java Desktop System<sup>65</sup>, NeWS<sup>66</sup>...

(Takes a deep breath, shakes his head, keeps writing.)

Imagine going to a McDonald's (another pillar of our capitalist world) and similarly being asked a myriad of questions before ordering a Happy Meal. You would run away in despair; yet, this is the reality of the Linux galaxy, where for better or worse, choice is king.

So much for communism<sup>67</sup>, am I right Steve?

The reality is that most users in 2026 compute very happily with a relatively recent 64-bit Intel-based laptop, with 8 or 16 GB of RAM and a 512 GB hard disk, probably formatted with Btrfs, bundled with the GNOME or KDE desktop environment on Wayland, using systemd, consuming software from either the APT or RPM package managers, including Flatpak for a more end-user-friendly interface to install userland applications, and with LibreOffice and Firefox pre-installed, thank you so much. And that means in general Ubuntu<sup>68</sup> (or any of its variants, like Zorin OS<sup>69</sup>) or Fedora<sup>70</sup> (for slightly more advanced users).

If you are into building containers, we cannot recommend Alpine<sup>71</sup> too much.

And if you are a really, really, really advanced user, particularly of the paranoid kind, some mix of Arch<sup>72</sup> or Qubes OS<sup>73</sup> will do just fine. We have mentioned the latter in a previous article about Joanna Rutkowska<sup>74</sup>:

*She is also the founder and main developer behind Qubes OS, described as a “reasonably secure operating system”, and praised by none less than Edward Snowden himself.*

Finally, for the French government officials reading these lines in quest of enlightenment, you might want to consider using SUSE<sup>75</sup>, RHEL<sup>76</sup>, or even some business-friendly versions of Ubuntu. (Disclaimer: I am currently working for the company that makes RHEL and Fedora.)

There you go, I have just solved your Linux overchoice in a couple of paragraphs. You are welcome.<sup>77</sup> If all else fails, you can even test individual distros directly from the comfort of your browser<sup>78</sup>, no need to install anything on your laptop.

Cover photo by □□□ Yumu<sup>79</sup> on Unsplash<sup>80</sup>.

## REFERENCES

- <sup>1</sup> [https://en.wikipedia.org/wiki/Non-Aligned\\_Movement](https://en.wikipedia.org/wiki/Non-Aligned_Movement)
- <sup>2</sup> [https://en.wikipedia.org/wiki/The\\_End\\_of\\_History\\_and\\_the\\_Last\\_Man](https://en.wikipedia.org/wiki/The_End_of_History_and_the_Last_Man)
- <sup>3</sup> <https://www.zdnet.com/article/france-leaves-windows-for-linux-desktop/>
- <sup>4</sup> <https://deprogrammaticaipsum.com/on-the-duopoly-of-mobile/>
- <sup>5</sup> <https://en.wikipedia.org/wiki/Overchoice>
- <sup>6</sup> [https://en.wikipedia.org/wiki/Alvin\\_Toffler](https://en.wikipedia.org/wiki/Alvin_Toffler)
- <sup>7</sup> [https://en.wikipedia.org/wiki/Future\\_Shock](https://en.wikipedia.org/wiki/Future_Shock)
- <sup>8</sup> [https://en.wikipedia.org/wiki/The\\_Paradox\\_of\\_Choice](https://en.wikipedia.org/wiki/The_Paradox_of_Choice)
- <sup>9</sup> [https://en.wikipedia.org/wiki/Mac\\_OS\\_X\\_Server](https://en.wikipedia.org/wiki/Mac_OS_X_Server)
- <sup>10</sup> <https://deprogrammaticaipsum.com/michael-kerrisk/>
- <sup>11</sup> [https://en.wikipedia.org/wiki/Red\\_Star\\_OS](https://en.wikipedia.org/wiki/Red_Star_OS)
- <sup>12</sup> <https://www.deepin.org/index/en>
- <sup>13</sup> <https://www.extix.se/>
- <sup>14</sup> <https://www.pardus.org.tr/>
- <sup>15</sup> <https://peropesis.org/>
- <sup>16</sup> <https://redcorelinux.org/>
- <sup>17</sup> <https://mxlinux.org/>
- <sup>18</sup> <https://www.kumander.org/>
- <sup>19</sup> <https://blendos.co/>
- <sup>20</sup> <https://www.linuxliteos.com/>
- <sup>21</sup> <https://www.modiciaos.cloud/>
- <sup>22</sup> <https://www.winux.is/>
- <sup>23</sup> <http://www.ututo.org/>
- <sup>24</sup> [https://en.wikipedia.org/wiki/List\\_of\\_Linux-supported\\_computer\\_architectures](https://en.wikipedia.org/wiki/List_of_Linux-supported_computer_architectures)
- <sup>25</sup> [https://deprogrammaticaipsum.com%5Bpicks%5D\(https://help.ubuntu.com/community/SupportedArchitectures\)](https://deprogrammaticaipsum.com%5Bpicks%5D(https://help.ubuntu.com/community/SupportedArchitectures))
- <sup>26</sup> <https://www.winehq.org/>
- <sup>27</sup> <https://www.darlinghq.org/>
- <sup>28</sup> [https://en.wikipedia.org/wiki/APT\\_\(software\)](https://en.wikipedia.org/wiki/APT_(software))
- <sup>29</sup> <https://rpm.org/>
- <sup>30</sup> <https://pacstall.dev/>
- <sup>31</sup> <https://brew.sh/>
- <sup>32</sup> <https://www.gnu.org/software/libc/>
- <sup>33</sup> <https://musl.libc.org/>
- <sup>34</sup> <https://www.uclibc.org/>
- <sup>35</sup> <https://uclibc-ng.org/>
- <sup>36</sup> <https://en.wikipedia.org/wiki/Klibc>
- <sup>37</sup> <https://www.sourceware.org/newlib/>
- <sup>38</sup> <http://www.eglibc.org/home>

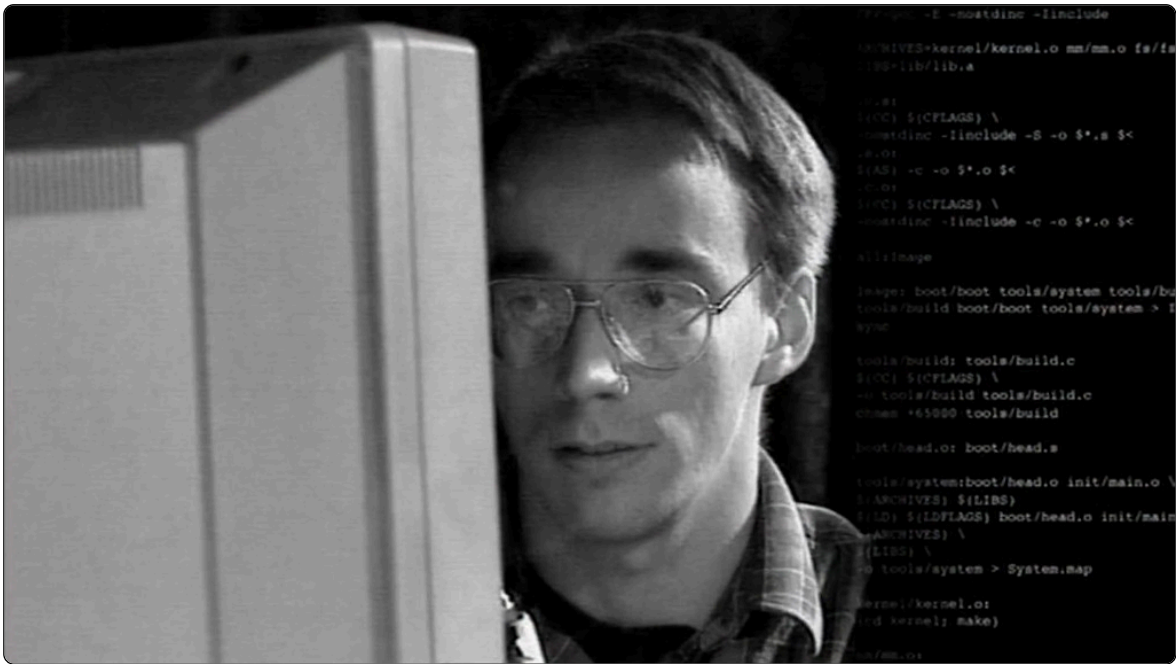
- <sup>39</sup> <https://www.fefe.de/dietlibc/>
- <sup>40</sup> [https://en.wikipedia.org/wiki/Bionic\\_\(software\)](https://en.wikipedia.org/wiki/Bionic_(software))
- <sup>41</sup> <https://www.gnome.org/>
- <sup>42</sup> <https://kde.org/>
- <sup>43</sup> <https://en.wikipedia.org/wiki/LXDE>
- <sup>44</sup> <https://xfce.org/>
- <sup>45</sup> [https://en.wikipedia.org/wiki/Common\\_Desktop\\_Environment](https://en.wikipedia.org/wiki/Common_Desktop_Environment)
- <sup>46</sup> <https://herbstluftwm.org/>
- <sup>47</sup> <https://i3wm.org/>
- <sup>48</sup> <https://github.com/baskerville/bspwm>
- <sup>49</sup> <https://www.enlightenment.org/>
- <sup>50</sup> <https://swaywm.org/>
- <sup>51</sup> [https://en.wikipedia.org/wiki/Cinnamon\\_\(desktop\\_environment\)](https://en.wikipedia.org/wiki/Cinnamon_(desktop_environment))
- <sup>52</sup> <https://fluxbox.org/>
- <sup>53</sup> <https://mate-desktop.org/>
- <sup>54</sup> <https://joewing.net/projects/jwm/>
- <sup>55</sup> <https://qtile.org/>
- <sup>56</sup> <https://en.wikipedia.org/wiki/Openbox>
- <sup>57</sup> <https://en.wikipedia.org/wiki/Blackbox>
- <sup>58</sup> <https://www.ctwm.org/index.html>
- <sup>59</sup> <https://fastestcode.org/>
- <sup>60</sup> <https://github.com/NsCDE/NsCDE>
- <sup>61</sup> <https://www.fvwm.org/>
- <sup>62</sup> <https://system76.com/cosmic/>
- <sup>63</sup> <https://en.wikipedia.org/wiki/Olwm>
- <sup>64</sup> <https://en.wikipedia.org/wiki/OpenWindows>
- <sup>65</sup> [https://en.wikipedia.org/wiki/Java\\_Desktop\\_System](https://en.wikipedia.org/wiki/Java_Desktop_System)
- <sup>66</sup> <https://en.wikipedia.org/wiki/NeWS>
- <sup>67</sup> [https://www.theregister.com/2000/07/31/ms\\_ballmer\\_linux\\_is\\_communism/](https://www.theregister.com/2000/07/31/ms_ballmer_linux_is_communism/)
- <sup>68</sup> <https://ubuntu.com/>
- <sup>69</sup> <https://zorin.com/os/pro/>
- <sup>70</sup> <https://fedoraproject.org/>
- <sup>71</sup> <https://www.alpinelinux.org/>
- <sup>72</sup> <https://archlinux.org/>
- <sup>73</sup> <https://www.qubes-os.org/>
- <sup>74</sup> <https://deprogrammaticaipsum.com/joanna-rutkowska/>
- <sup>75</sup> <https://www.suse.com/>
- <sup>76</sup> <https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux>
- <sup>77</sup> <https://deprogrammaticaipsum.com/contribute/>
- <sup>78</sup> <https://distrosea.com/>

<sup>79</sup> [https://unsplash.com/@cdd20?utm\\_source=unsplash&utm\\_medium=referral&utm\\_content=creditCopyText](https://unsplash.com/@cdd20?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)

<sup>80</sup> [https://unsplash.com/photos/a-group-of-red-arrows-on-a-black-surface-HQH-GOZ6K2c?utm\\_source=unsplash&utm\\_medium=referral&utm\\_content=creditCopyText](https://unsplash.com/photos/a-group-of-red-arrows-on-a-black-surface-HQH-GOZ6K2c?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText)



# Hannu Puttonen



By Adrian Kosmaczewski

Paraphrasing Apple and their famous advertising campaign<sup>1</sup>, we can safely say that Finnish movie director Hannu Puttonen (1960–2023) was very interested in the crazy ones. The misfits. The rebels. The troublemakers. The round pegs in the square holes. The ones who see things differently. He made a living telling their stories: those of fringe characters, edgy culture movements, and radical artistic outsiders. And that is why, in 2001, he directed the first documentary ever made about Linux, because if you think carefully about it, its story can be seen as a savant mixture of those three attributes.

Hannu Puttonen followed his gut instincts, and directed “Mr. Bragg Goes to Moscow”<sup>2</sup> (1988), his first documentary, in which followed British songwriter and left-wing activist Billy Bragg<sup>3</sup> on a concert tour of the Soviet Union during the Perestroika era. Because of course, why not<sup>4</sup>. In the same vein, came later “Bring

the Beat Back!”<sup>5</sup> (1992), one of the earliest documentaries to pay attention to the burgeoning techno scene, featuring pioneers like Bill Drummond (from The KLF<sup>6</sup>) and Richard H. Kirk (from Cabaret Voltaire<sup>7</sup>). This documentary was released well before Prodigy, Daft Punk, or Moby made electronica mainstream. Or last but definitely not least, “Momus Man of Letters”<sup>8</sup> (1994), an experimental documentary about the eccentric Scottish musician Nicholas Currie, aka Momus<sup>9</sup>, featuring appearances by Jarvis Cocker of British britpop legend Pulp<sup>10</sup>.

Still reading? If any of the names in the previous paragraph rang any bell in you without having to click to read their Wikipedia entries, it means that you are, just like the author of this article, a proud member of Generation X.

Hannu Puttonen was, then, the perfect filmmaker for the first documentary about Linux, and as explained above, not only because he was Finnish like Linus Torvalds himself. So came to be “The Code”<sup>11</sup> (2001), a very fittingly hour-long documentary that we have decided to feature as this month’s Vidéotheque entry.

By the time Puttonen interviewed Linus, the Kernel (in uppercase, please) was barely a decade old, and yet it was gently starting to stir the pot of technology economics. For the sake of narration, the early story of the Kernel is then split in three major sections: birth, philosophy, and economics.

First and foremost, Linux was born a true hobby project. We all know that by now. But back in 1991, having an operating system developed via the collaboration of developers (or “worker ants”<sup>12</sup>) spread all over the world was unprecedented, radically subversive, and fascinating. Of course, the documentary explains around minute seven<sup>13</sup> that “Linux” the kernel is not the same as “Linux” the operating system.

Then comes the philosophy. As I wrote in a previous article<sup>14</sup> of this magazine,

*Everything changed in 1998. That is when Christine Peterson coined the phrase “Open Source,” deliberately masking the business-unfriendly moniker of “Free Software” which Wall Street abhorred so much. More or less at the same time, Netscape decided to rewrite its own product. Such wise decision, widely celebrated by the “community” helped Microsoft win the infamous browser*

*wars; this situation ultimately begat Internet Explorer 6 and its horrible crush on web development for the next decade.*

Richard Stallman has spent countless hours for the past 30 years explaining people that Free Software ≠ Open Source, but few people are paying attention. This, also, is explained in the documentary.

Microsoft took notice and decided to attack the whole concept. Hence the final part of “The Code”, telling us about the economics and dynamics of the Linux Kernel project. From Torvalds’ “Benevolent Dictator” status, to the “dot com boom” of the late 1990s, and even showing an interview of Bob Young, one of the founders of Red Hat around minute 38<sup>15</sup>. It all ends with a nice bow, predicting the impact of Linux in emerging markets and its impact in software intellectual property, a never ending issue, around minute 48<sup>16</sup>.

Oh, and our preferred quote, from legendary Kernel maintainer Alan Cox<sup>17</sup> around minute 24<sup>18</sup>:

*To me code has more in common with for example poetry or some kinds of writing. The beauty of it is in the structure, in putting ideas across one at a time in a clear way. So a good piece of code you read without comments and it's immediately obvious why it's been written, how it's elegant. So you're looking for code which is both clean and elegant. But also doesn't rely on clever programming tricks, doesn't make assumptions which may not be true in the future. Because the last thing we want to do is having much code in the Linux kernel which requires large amounts of effort to keep it working. We want code which will just continue to work, and work forever.*

This film having been released in 2001, no mentions ~~can~~ should be found therein about “The Cloud”, Android, containers<sup>19</sup>, Kubernetes, NASA’s Ingenuity<sup>20</sup>, Valve’s Steam Deck<sup>21</sup>, cars<sup>22</sup>, or other interesting things people have built upon Linux during the first quarter of the 21st century. On the other hand, TiVo was around at the time of the release of this film, it was already using Linux, and not without controversy<sup>23</sup>.

Internet wisdom usually advocates for *not reading comments*, but I feel the need to reproduce one from YouTube user Peter Trinh<sup>24</sup>, which faithfully conveyed the spirit of the documentary, resonating ideas around a “decentralized economic systems based on mutual aid and voluntary cooperation” we talked about in a previous article<sup>25</sup>:

*I just saw this documentary. I really enjoyed it. I loved the passion and beliefs of people that showed through. This is the kind of stuff that enamored me to computers and Linux in the first place. A sort of innocence, curiosity, and a deep desire to do something good, and doing the work for a vague something-something that is larger than us. So much that we would be willing to do this lovingly as a hobby. I really miss the days like this when we could share this together, even across distances and mindsets.*

There is nothing preventing us from bringing those days and those feelings back into fashion, other than our own will, that is.

Watch this month’s Vidéothèque entry, “The Code” (2001), by Hannu Puttonen, on YouTube<sup>26</sup>.

(Oh, and if you are fluent in Finnish and cannot get enough computer history in your system, we can recommend watching the launch<sup>27</sup> of Linux 1.0 on March 30th, 1994, by Linus Torvalds himself. You are most welcome.)

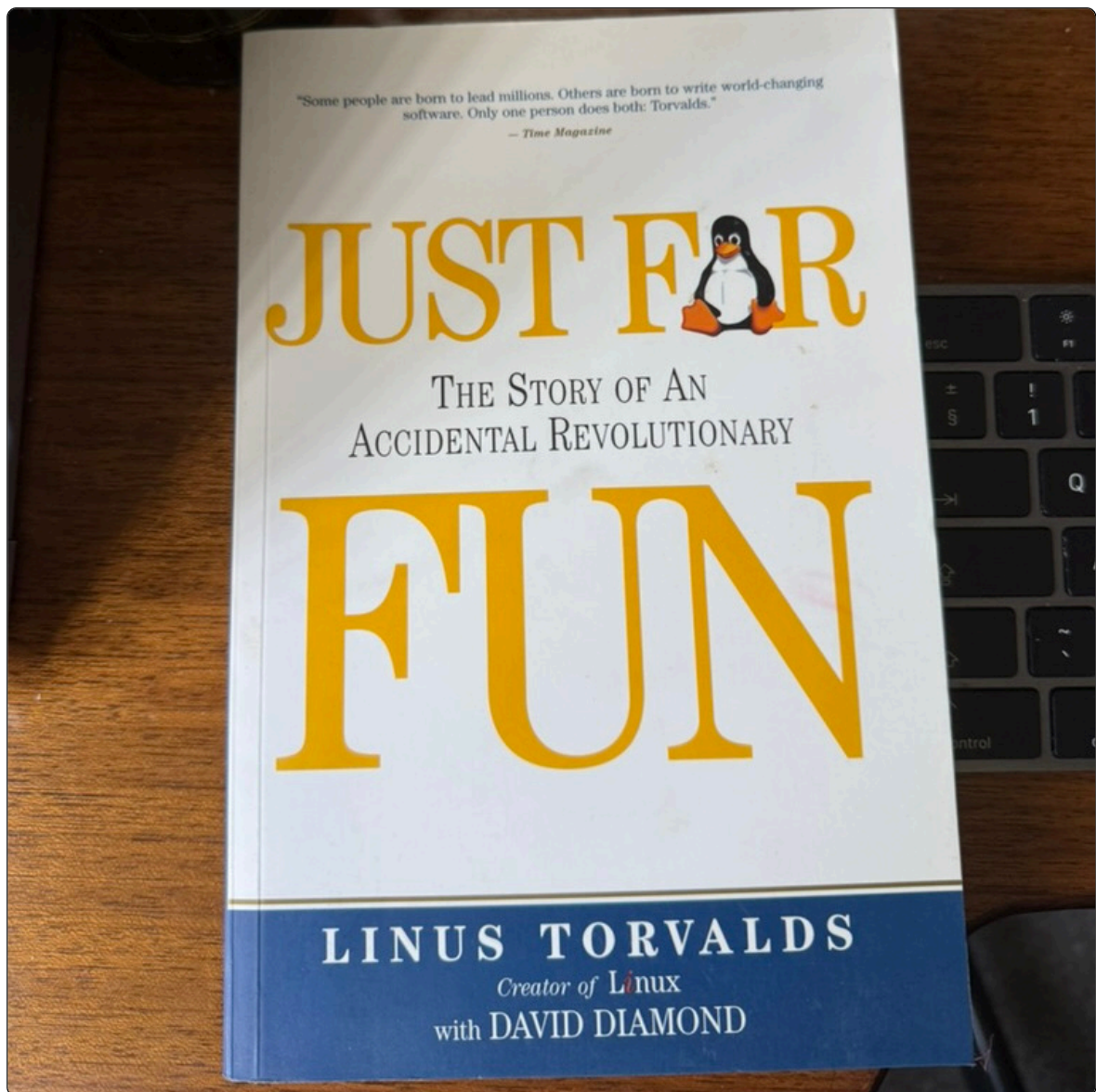
Cover snapshot chosen by the author.

## REFERENCES

- <sup>1</sup> [https://en.wikipedia.org/wiki/Think\\_different](https://en.wikipedia.org/wiki/Think_different)
- <sup>2</sup> <https://www.imdb.com/title/tt0410339/>
- <sup>3</sup> [https://en.wikipedia.org/wiki/Billy\\_Bragg](https://en.wikipedia.org/wiki/Billy_Bragg)
- <sup>4</sup> <https://www.youtube.com/watch?v=Q8v5dYH0iQ>
- <sup>5</sup> <https://www.imdb.com/title/tt0233381/>
- <sup>6</sup> [https://en.wikipedia.org/wiki/The\\_KLF](https://en.wikipedia.org/wiki/The_KLF)
- <sup>7</sup> [https://en.wikipedia.org/wiki/Cabaret\\_Voltaire\\_\(band\)](https://en.wikipedia.org/wiki/Cabaret_Voltaire_(band))
- <sup>8</sup> <https://www.imdb.com/title/tt0410330/>
- <sup>9</sup> [https://en.wikipedia.org/wiki/Momus\\_\(musician\)](https://en.wikipedia.org/wiki/Momus_(musician))
- <sup>10</sup> [https://en.wikipedia.org/wiki/Pulp\\_\(band\)](https://en.wikipedia.org/wiki/Pulp_(band))
- <sup>11</sup> <https://www.imdb.com/title/tt0315417/>
- <sup>12</sup> [http://www.youtube.com/watch?v=zPt\\_e9Cdk08&t=244](http://www.youtube.com/watch?v=zPt_e9Cdk08&t=244)
- <sup>13</sup> [http://www.youtube.com/watch?v=zPt\\_e9Cdk08&t=421](http://www.youtube.com/watch?v=zPt_e9Cdk08&t=421)
- <sup>14</sup> <https://deprogrammaticaipsum.com/open-always-wins/>
- <sup>15</sup> [http://www.youtube.com/watch?v=zPt\\_e9Cdk08&t=2292](http://www.youtube.com/watch?v=zPt_e9Cdk08&t=2292)
- <sup>16</sup> [http://www.youtube.com/watch?v=zPt\\_e9Cdk08&t=2906](http://www.youtube.com/watch?v=zPt_e9Cdk08&t=2906)
- <sup>17</sup> [https://en.wikipedia.org/wiki/Alan\\_Cox\\_\(computer\\_programmer\)](https://en.wikipedia.org/wiki/Alan_Cox_(computer_programmer))
- <sup>18</sup> [https://www.youtube.com/watch?v=zPt\\_e9Cdk08&t=1448s](https://www.youtube.com/watch?v=zPt_e9Cdk08&t=1448s)
- <sup>19</sup> <https://deprogrammaticaipsum.com/shipping-your-computer/>
- <sup>20</sup> [https://en.wikipedia.org/wiki/Ingenuity\\_\(helicopter\)](https://en.wikipedia.org/wiki/Ingenuity_(helicopter))
- <sup>21</sup> <https://en.wikipedia.org/wiki/SteamOS>
- <sup>22</sup> <https://arstechnica.com/cars/2024/04/linux-is-now-an-option-for-safety-minded-software-defined-vehicle-developers/>
- <sup>23</sup> <https://en.wikipedia.org/wiki/Tivoization>
- <sup>24</sup> <https://www.youtube.com/@Thepilgrimingtrinh>
- <sup>25</sup> <https://deprogrammaticaipsum.com/the-conquest-of-code/>
- <sup>26</sup> [https://www.youtube.com/watch?v=zPt\\_e9Cdk08](https://www.youtube.com/watch?v=zPt_e9Cdk08)
- <sup>27</sup> <https://www.youtube.com/watch?v=Pd3P-68at9E>



# Linus Torvalds & David Diamond



By Graham Lee

Many autobiographies of famous people involve a certain amount of ghostwriting, if the subject and alleged author is not a professional writer. An actual writer listens to their stories, interviews them, maybe gets them to draft some anecdotes or chapters, then works all of that up into a narrative that a potential audience might consider readable, all the while trying to maintain some sense of the “authentic” subject’s voice. It is common for that ghostwriter to get a byline on the cover, as in “iWoz” by Steve Wozniak with Gina Smith, or “Under the Radar” by Robert Young and, in smaller letters, Wendy Goldman Bohm. Sometimes, you have to hunt around to work out who the ghostwriter is, as in “Lean In” by Sheryl Sandberg (and Nell Scovell, who may have done well to read the message in that very book about getting a seat at the table).

When it comes to Linus Torvalds’ autobiography, “Just for Fun: the story of an accidental revolutionary”<sup>1</sup>, author and technology journalist David Diamond encountered a significant problem: Torvalds could not remember many of the details of his life, and was not too interested in recounting them. This even though Torvalds *is* a professional writer, who has spent most of his career writing emails, Usenet posts, and commit messages to manage and document the development of the Linux (nee “Freax”) kernel. Of course, he is not always the most politically-adept writer of emails, so maybe it is best that he did not write this book on his own.

As such, “Just for Fun” is two books for the price of one. Most of the chapters adopt Torvalds’ voice, giving us an autobiography that explains such of his life as he can recall. That turns out to be very little that is not about computers, and a lot about the explosive growth of Linux alongside open source business models, and the people he met while working on Linux. Luckily for those of us interested in technology (and specifically, those of us writing a magazine issue about Linux), the latter part is incredibly detailed.

We learn that Linux started life as a terminal emulator that Torvalds wrote to learn more about the 386 processor. He wrote the terminal emulator in the Minix operating system, but it ran as a standalone program that spoke directly to the PC BIOS. As the terminal emulator grew in complexity, Torvalds began to realise that he was writing a simple, shadow operating system and focused on implementing

that in a POSIX-compatible way. The rest, as they say, is history (along with quite a few arguments and flame wars).

We get to learn that Torvalds met Bill Joy twice, for example, and walked out on him when Sun described their approach to JINI as open source but he discovered that it was not. He merely met Steve Jobs once, who tried to convince Torvalds to give up on Linux and join Apple on making Mac OS X. As you can probably tell from the way Torvalds stayed at Transmeta<sup>2</sup> and now works for the Linux Foundation<sup>3</sup>, the reality distortion field was not operating at full intensity on that day.

In between these chunks of the book, in italicised tones that try to impart a matter-of-fact, “this is the life of a jobbing ghostwriter” vibe to what must have been an increasingly stressful experience, Diamond’s voice breaks through. In these sections, the book flips from an autobiography to a meta-biography, in which Diamond details the conversations he had with Torvalds, family members, and others, to get the information about Torvalds that is not in the autobiographical sections. We see the life of someone who is writing a life of someone else.

Cover photo by the author.

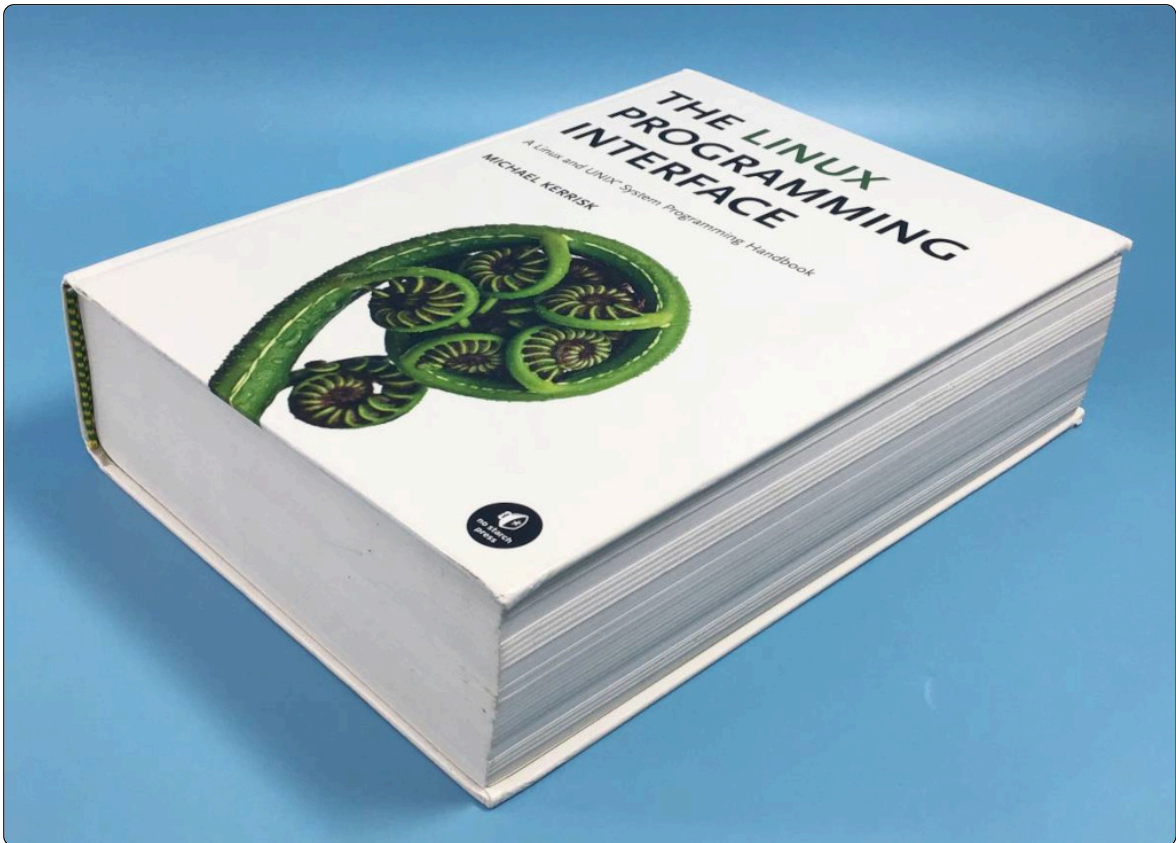
REFERENCES

<sup>1</sup> <https://archive.org/details/justforfun00linu>

<sup>2</sup> <https://en.wikipedia.org/wiki/Transmeta>

<sup>3</sup> <https://www.linuxfoundation.org>

# Michael Kerrisk



By Adrian Kosmaczewski

There was a time before eBooks, when developers had to buy actual massive paper editions of the most precious titles in their craft, and in some case, they had to carry them around, either for work or (also possible) for pleasure. In this particular case I have to say that I am happy to own the digital version of this Library entry, because as you can see on the cover picture of this article, it is a massive book.

We are talking about the 2010 book “The Linux Programming Interface: A Linux and UNIX System Programming Handbook”<sup>1</sup> by Michael Kerrisk<sup>2</sup>, a massive title with 1552 pages of incredible detail and relevance, available from No Starch Press<sup>3</sup> at the time of this publication.

Let us be honest. The fact that this book has its own Wikipedia entry<sup>4</sup> says a lot more than I could in this article; let us try to summarize (somehow) this massive volume in less than a thousand words.

Michael Kerrisk, a New Zealand native living in Germany, is a very well-known name in the Linux world. If you have ever used the `man` utility to read manual pages in your Linux system you have most certainly already read his writing: Michael has authored hundreds of “man pages”, and in some way it feels like writing this book was just inertia at work.

The book title explains that the scope of the book is the “programming interface” of the operating system; this description encompasses both the system calls *and* the GNU C standard library (also known as `glibc` in programming circles). The book dives into both, not separately, but at the same time, as it advances topic by topic (and there are 64 chapters filled with those).

The most interesting thing about this book is precisely that it is not just a catalog of functions with their description (we have the manual pages for that, after all). Each chapter is dedicated to a particular domain of interest: files, processes, threads, libraries, networking, user management, and so on and so forth. Each topic receives a very careful treatment, including edge cases, code examples, and even exercises at the end.

In a previous article<sup>5</sup> we have talked about Abraham Silberschatz, Peter Baer Galvin, & Greg Gagne’s “Operating System Concepts” coursebook, a common choice in many bachelor programs, as it offers a wide overview of concepts; however, for students interested in the internals of the Linux operating system (I would say for those in a Master or PhD program) Kerrisk’s book will become second nature, and a mandatory read.

Just a warning to my readers; this book is not a Linux manual, so if you are looking for help about how to install or use any Linux distribution, you will be better served someplace else. The introduction, however, includes some interesting historical details about UNIX, Linux, and the C programming language, which might be most interesting to any programmer interested in the platform.

But if you happen to write any kind of software for Linux that needs to interact with the operating system internals in any way, in particular if you are using C, there is definitely no better book to read than this one. In that case, we would suggest complementing this book with the original Michael K. Johnson's 1995 paper "The Linux Kernel Hackers' Guide"<sup>6</sup>; Rusty Russell's "Unreliable Guide To Hacking The Linux Kernel"<sup>7</sup> (2000); and even with the original programmer's manual: Ken Thompson & Dennis Ritchie's own "Unix Manual, first edition"<sup>8</sup> originally published in November 1971. Oh, and for history boffins, get yourself a copy of the "Linux Kernel History Report 2020"<sup>9</sup> published by the Linux Foundation.

Cover photo from a Yahoo! Japan auction<sup>10</sup>.

## REFERENCES

<sup>1</sup> <https://man7.org/tlpi/>

<sup>2</sup> <https://www.man7.org/mtk/index.html>

<sup>3</sup> <https://nostarch.com/tlpi>

<sup>4</sup> [https://en.wikipedia.org/wiki/The\\_Linux\\_Programming\\_Interface](https://en.wikipedia.org/wiki/The_Linux_Programming_Interface)

<sup>5</sup> <https://deprogrammaticaipsum.com/abraham-silberschatz-peter-baer-galvin-greg-gagne/>

<sup>6</sup> [https://github.com/trimstray/technical-whitepapers/blob/master/base/the\\_linux\\_kernel\\_hackers\\_guide.pdf](https://github.com/trimstray/technical-whitepapers/blob/master/base/the_linux_kernel_hackers_guide.pdf)

<sup>7</sup> <https://www.kernel.org/doc/html/latest/kernel-hacking/hacking.html>

<sup>8</sup> <https://www.nokia.com/bell-labs/about/dennis-m-ritchie/1stEdman.html>

<sup>9</sup> <https://www.linuxfoundation.org/resources/publications/linux-kernel-history-report-2020>

<sup>10</sup> <https://auctions.yahoo.co.jp/jp/auction/x1106003396>