Linus Torvalds's Lessons on Software Development Management

Share

by sjvn01 26-09-2011 10:14 AM - edited 28-09-2011 06:50 AM

If anyone knows the joys and sorrows of managing software development projects, it would be Linus Torvalds, creator of the world's most popular open-source software program: the Linux operating system. For more than 20 years, Torvalds has been directing thousands of developers to improve the open source OS. He and I sat down to talk about effective techniques in running large-scale distributed programming teams – and the things that don't work, too.

Torvalds says there are two things that people very commonly get completely wrong, both at an individual developer level and at companies.

"The first thing is thinking that you can throw things out there and ask people to help," when it comes to open-source software development, he says. "That's not how it works. You make it public, and then you assume that you'll have to do all the work, and ask people to come up with suggestions of what you should do, not what they should do. Maybe they'll start helping eventually, but you should start off with the assumption that you're going to be the one maintaining it and ready to do all the work."

Torvalds continues, "If you start off with some 'kumba-ya feeling' where you think people from all the world are going to come together to make a better world by working together on your project, you probably won't be going very far."

"The other thing—and it's kind of related—that people seem to get wrong is to think that the code they write is what matters," says Torvalds. Most software development managers have seen this one. "No, even if you wrote 100% of the code, and even if you are the best programmer in the world and will never need any help with the project at all, the thing that really matters is the users of the code. The code itself is unimportant; the project is only as useful as people actually find it."

I'll add at this point that this isn't just a programmer problem. I've seen entire companies get locked into the idea that "perfecting" the program was everything. They then neglected what the users wanted from the program, supporting the users and so on. Most of us who've been in the business for a while have seen this cycle play out over and over again.

Expanding on that second point, Torvalds says that's why the Linux kernel team is "so very anal about the whole 'no regressions' thing, for example. Breaking the user experience in order to 'fix' something is a totally broken concept; you cannot do it. If you break the user experience, you may feel that you have 'fixed' something in the code, but if you fixed it by breaking the user, you just violated that second point; you thought the code was more important than the user. Which is not true."

Torvalds concludes, "Way too many projects seem to think that the code is more important than the user, and they break things left and right, and they don't apologize for it, because they feel that they are 'fixing' the code and doing the right thing."

To that I can only add "Amen!"

On the Importance of Development Tools

I also asked Torvalds about Software Configuration Management (SCM) tools like his own Git version control system. He replied, "I don't think tools are all that fundamentally important."

"Now, what is important is that there's a good workflow for the project, and tools can

Top Tags

Security management mobile innovation career hardware cloud computing networking trends hiring printing workflow data center green infographic budget collaboration geek social media marketing storage big data power roi Linux nostalgia virtualization administration conference smartphone

View All

The HP Input Output site is sponsored by HP and features articles and content from HP and third-party contributors. Third-party articles and content, while paid for by HP, do not necessarily represent the views and opinions of HP. HP does not endorse this content and is not responsible for its accuracy, availability and quality.

Follow Us

Spotlight

"It's Not My Job" - Handling the Vendor Finger-Pointing Trap

Category Browser

- Input Output
  - Feature Articles
  - The Green Horizon
  - Mobility Matters
  - Security, the red-haired step child of IT
  - Policy Watch
  - Active Information
  - HPIO Video
  - Clearing Up The Cloud

certainly help with that," said Torvalds. "But most projects don't necessarily really need tools. There's a lot of projects that simply don't have enough changes to really require any tools at all for their work flow; if you only have a few hundred patches per release, you can maintain those just about any way you want, including entirely by hand."

Linux is a different story of course. "For the kernel, we have thousands of patches flying around every release, and a release roughly every three months, and so for us the tools really are very important," he says. "But I still don't think it was all that big a mistake to just do tar-balls and patches for the first few years of development; it was a much smaller project back then, and it took several years for the lack of tools to really become a problem."

Besides, "Some tools encourage workflows that are actively detrimental, and I think CVS [Concurrent Versions System, a version control system] for example has caused a lot of projects to have the notion of a 'commit cabal,'" Torvalds continues. "I personally tend to think tar-balls and patches are actually preferable to that – if only because they make all developers 'equal,' and you don't get the kind of model where certain people have 'commit access,' and the rest are second-class citizens. Sometimes it's better that everybody is a second class citizen than that some people have an easier time at it."

> Torvalds, I should note, knows CVS well and has hated it for years. As he said in a Google Talk in 2007, "I hate CVS with a passion."

Torvalds continues, "Much more important than the tools is the people. The maintainers, and the mindset."

Keeping People On Track

And how do these people work together today? I asked Torvalds about the role of the Linux Kernel Mailing List (LKML) in the process. He replied, "I think Linux used to 'happen' more on LKML than it does these days. The signal-to-noise ratio and just the pure volume of LKML means that most developers simply don't have time to really read LKML—at best they scan subject lines. As a result, these days I'd argue that most of the real development happens within the sandbox of single developers, and then email on more of a person-to-person scale is actually how things really get done."

That said, "That doesn't mean that LKML isn't important; it means that LKML has become the 'public band' of all those individual email threads," Torvalds adds. "So what ends up happening is that you have maybe four or five people involved in a discussion about their work, but LKML stays cc'd on the whole thing. That turns what would otherwise be a purely private discussion into something where others can jump in."

Here's how it works, "A lot of people actually don't really 'read LKML;' they often auto-archive it, but then react to certain keywords or, more often, [to] key people being involved in the discussion."

"It also acts as a kind of archiving notion," Torvalds continues, "so that people can refer to it later, and a lot of bug reports end up being found by Googling for them. If somebody raises an issue, it may well be some odd hardware problem, but if Google shows that it's been raised several times on LKML in the past, that starts to indicate that it may be obscure, but it's certainly not some totally isolated issue."

"So I think LKML is really quite important, but no, it's not how we keep people 'on track,'" he says. "All the developers tend to be pretty self-motivating, and they all have sane ideas (well, the core ones do by definition – because that's how they became core developers, by showing that they had good taste and high motivation). It's important simply because that 'public part' of the discussions are still important, even if in practice it's often a pretty small core in any particular discussion. Things are simply different when they happen in the open," concludes Torvalds.

On Delegating – and Staying Sane

Once, Linux was a solo project. It now has thousands of committers and contributors. I then asked, "How much delegating do you these days? Any thoughts on how to delegate to keep one's sanity and the workflow flowing?"

"If there's one thing I've learnt, it is that you have to learn to let go and not try to control people and the code," he says. "If you don't think somebody else can do it on their own without your oversight, you might as well give up immediately as a maintainer."

He continues, "Yes, I often get involved in small details, but it's not because I don't trust people or don't delegate. It's because some small detail ends up being brought up to me. Either it's a bug (and they are almost all just silly small details that got overlooked), or it's

just some workflow issue that bothers me (like me complaining about the developer names not showing up properly in the logs earlier today to one sub-maintainer)."

Still, says Torvalds, "Those details have to be occasional details, not the kind of 'look over the shoulder of the developer to check everything he does.' I trust sub-maintainers to do the right thing 99% of the time. And then, very occasionally, I end up complaining loudly about something." Say, for example, on how the open-source GNOME desktop is, or rather isn't, moving forward.

So, there you have it. That's some of the ways Torvalds does it. And, if you think you know better, ask yourself: Have I created a world-class operating system that runs most supercomputers, stock-exchanges, and websites like Google? If your answer's no, I'd re-read his answers and take a long hard think about how you've been managing your own projects.

Everyone's Tags: collaboration Linux management open source software development View All (6)

40 Comments (40 New) Permalink View Article Reactions

Article Options

Comments

by Tasha(anon) on 26-09-2011 12:31 PM Options

What! No audio/video stream? I aint reading all that! ;D

Permalink

by maarten(anon) on 26-09-2011 12:31 PM Options

At the risk of being pedantic, I would like to point out the Linux is a kernel, not an operating system, as stated incorrectly in the article several times.

Permalink

by John Smith 123(anon) on 26-09-2011 12:50 PM Options

"What! No audio/video stream? I aint reading all that! ;D" Said the developer who never tries to improve how they develop.

Permalink

by James Broadhead(anon) on 26-09-2011 01:28 PM Options

"Have I created a world-class operating system that runs most supercomputers, stock-exchanges, and websites like Google?"

I can't believe that I'm *that guy*, but you are making the pretty common mistake of failing to distinguish between the Kernel of an Operating System, and the other components which are required. Linus didn't write it all by any measure of the imagination.

Permalink

by Richard Chapman(anon) on 26-09-2011 02:25 PM Options

Yes, Linus Torvalds is a unique and very talented individual. Why some people just can't bring themselves to admit that I'll never understand, kernel, operating system or pumpkin seed.

Permalink

by Michael McClary(anon) on 26-09-2011 02:28 PM Options

Regarding Kernel vs. OS: Back in the early days what we now call a Kernal WAS called the OS. Including all the maintenance utilities, d(a)emons, and major apps in the definition of OS is more recent. (I presume this revised definition got its start when things like file systems started moving out of supervisor space into applications with special privileges or communication hooks in the kernel.) I'm partial to including only the boot mechanism, supervisor/scheduler, major drivers, maybe the file system, and any permanently loaded utilities and command processors in the "OS" definition, with loadable command drivers and utilities (both user and maintenance) outside, and loadable drivers in limbo. But that's because I grew up with the older definition. IMHO the packaged utilities (including the window system), should be OUTSIDE the OS definition (both old and new), even if they're standard and included with all distributions. This is particularly true with unix/linux, where most of the maintenance utilities are actually just GUI or syntactic sugar (with sanity checks) specialized editors, replacing manual editing of (usually human-readable) configuration files. Rule of thumb for the older, tighter definition: If it's not in chapters 2, 4, 9, or part of 5 of the help command (and it's not "init") it's a utility or library, not part of the OS (no matter HOW central, fundamental, or necessary).

Permalink

---

by chakkerz(anon) on 26-09-2011 02:33 PM Options

Isn't this a form of association fallacy. Linus wrote a great kernel. The development was Project Managed. Therefore Linus knows all about Project Management.

That is not to say he doesn't know about project management, but the quality or widespread use of one product doesn't say anything about its development process. <Insert cheap shot about Windows/Microsoft, if required>

Permalink

---

by akohlsmith(anon) on 26-09-2011 02:48 PM Options

"What! No audio/video stream? I aint reading all that! ;D"

I see the smiley there but an A/V link so you don't "waste your time" ? Really? I find text SO MUCH better than video or audio streams. I can skim/re-read and othewise really get the content out of the text, much more easily than I can with a video stream.

In fact I think that's one of the biggest problems with video or audio streams... they require much, much more of my brain to extract the meaning, and skipping about in them is much harder as well.

Permalink

---

by pinkboi(anon) on 26-09-2011 03:07 PM Options

...creator of the world's most popular open-source software program: the Linux operating system.

While other people are being pedantic about OS vs. kernel, I should correct the error in the *first sentence* - there were many open source projects before Linux came around! Sure, Linux has done much to bring awareness of opensource to the general public, but c'mon. It was compiled with gcc, an opensource project that started several years before Linux.

Permalink

---

by Otto(anon) on 26-09-2011 03:12 PM Options

Looks like you're running Sharepoint. :(I get this error from the links -> "An Unexpected Error has occurred."

Posting a Linus Torvalds interview on some crappy Microsoft based web site is kind of insulting, don't you think?

Permalink

---

by Tim(anon) on 26-09-2011 04:48 PM Options

...creator of the world's most popular open-source software program: the Linux operating system.

While other people are being pedantic about OS vs. kernel, I should correct the error in the first sentence - there were many open source projects before Linux came around! Sure, Linux has done much to bring awareness of opensource to the general public, but c'mon. It was compiled with gcc, an opensource project that started several years before Linux.

I think your mixing up 'most popular' with 'first'.

The arguement about whether it is the 'most popular' or not can still be made, and the answer can depend on whether you measure 'most poular' as 'most widely used' or 'most widely beloved' - given that both Firefox and Eclipse can run on windows they can claim additional users that aren't running or like Linux.

Permalink

---

by kj(anon) on 27-09-2011 02:09 AM Options

I liked this article. It got to some very interesting macro points that need to be reflected on to get. Reading it slowly helps with this. Thanks.

Permalink

---

by Peter(anon) on 27-09-2011 02:41 AM Options

"but you are making the pretty common mistake of failing to distinguish between the Kernel of an Operating System"

That is not a fail, that is common sense.

Permalink

---

by darkrose(anon) on 27-09-2011 03:37 AM Options

While a few people have rightly pointed out that Linux is a kernel, not a whole operating system (even Michael McClary's definition puts it as a kernel, not an OS), no one has pointed out that the Linux is not open source, it is Free Software, which is not the same thing at all.

http://www.gnu.org/philosophy/open-source-misses-the-point.html

Permalink

---

by Fadi(anon) on 27-09-2011 05:26 AM Options

Unfortunately, not all developers out there are like Linus. They can't project manage themselves, and they can't create (probably the best) operating system by themselves.

In this day and age, there are many reasons why projects fail, and most of them have either to do with the fact that the project is not managed by the same person who solely works on it, see: http://www.pmhut.com/why-projects-fail-2

Permalink

---

by Peter(anon) on 27-09-2011 06:09 AM Options

That is corporate made software, see:
this article for some more experience.

Permalink

---

by ADMIN **estherschindler** on 27-09-2011 06:45 AM Options

Geez, folks. Is kernel versus OS the only thing you got out of the article? Nothing about, say, the opinions of someone who has been intimately involved in managing a large, complex, international project that has opinionated contributors (many of whom are wholly volunteers and thus cannot be motivated by "If you don't code, you don't eat), software that *must* be of the highest quality, and all the other unique (and not-so-unique) issues of project management?

--Esther, the site editor

Permalink

---

by SAB(anon) on 27-09-2011 09:01 AM Options

Geez, folks. Is kernel versus OS the only thing you got out of the article?

To be fair to the people arguing about the kernel/os differences, the author pretty much pre-slammed any objective discussions about the article when he poses the very insulting rhetorical question:

And, if you think you know better, ask yourself: Have I created a world-class operating system...

There are plenty of brilliant people in this world that have *not* created a world-class operating system, and of the ones that have, they probably feel as though they have better things to do than to waste their time reading this interview, especially given how argumentative Linus has been with them in the past.

From this quote page you can see how he feels about things that don't toe the line of his opinion:

"My personal opinion of Mach is not very high. Frankly, it's a piece of crap. It contains all the design mistakes you can make, and even managed to make up a few of its own."

My guess is that the people that work on those kernels are not going to take much stock in what Linus is saying and thus probably not bother to read this article.

Permalink

---

by Loke(anon) on 27-09-2011 10:21 AM Options

Someone needs to explain to Linus, that the user experience and code management have nothing to do with each other, and if they do, and the management has failed.

It's like fixing the generator, and expecting the user would be upset, because his appliances may stop working. I like the first point though.

Permalink

---

by udnaan(anon) on 27-09-2011 11:41 AM Options

No one gives a **bleep** about the great project management tips that Linus gave out for free. Nope, not at all. What is more important is to point out that Linux is a kernel (seriously? If you think this is even worth raising, then you are in the wrong field, consider switching to plumbing) or that Linux is not "FREE" software.

Permalink

---

by silent sojourner(anon) on 27-09-2011 12:43 PM Options

People just want to show others that they themselves are right, rather than admitting that others are right.

Permalink

---

by irisguyaer(anon) on 27-09-2011 09:56 PM Options

I learned so many lessons out from this article like "users are more important than your code" and "you need to ask people what things you should do to your project, and NOT telling them that _they_ need to do this and that to your project."

Then the commenters shouted all along with "Not open source, but free software" and "OS vs. Kernel" thingy.

And SAB, are you an avid fan of GNOME Shell 3 to the point that it becomes a religious artifact to you? Take the lessons out from this article and debunk them if you do not like them and present a more reasonalbe approach instead of ad hominem. SJVN's asking of you people is that you need to ask yourselves how your projects going well, not necessarily that all of us will create great operating systems that will run anything under the sun.

Permalink

---

by jsz(anon) on 27-09-2011 11:51 PM Options

The link on "Say, for example, on how the open-source GNOME desktop is, or rather isn't, moving forward." is broken. Anyone knows where the link actually is?

Permalink

---

by peter(anon) on 28-09-2011 03:01 AM Options

also a very usefull comment:

"Linus didn't write it all by any measure of the imagination"

- really? Perhaps writing is not the only contribution to 'creating'.

another good one:

"Someone needs to explain to Linus, that the user experience and code management have nothing to do with each other, and if they do, and the management has failed."

- Perhaps somehow the tools and the results of a project have a connection?

The interview can learn you something about projects and project management, the reactions can learn you something about well, the commenters.

Permalink

---

by ADMIN **estherschindler** on 28-09-2011 06:52 AM Options

@jsz Thanks for the head's up. I fixed the link.

Permalink

---

by Barry Wey(anon) on 28-09-2011 11:58 AM Options

Linus Torvalds shows us how to do the RIGHT thing on software development.

I have to say he's one of the greatest programmer in this world. but it doesn't mean that he's correct eventually. yes i love Linux and i really like to do some work on that. look around and is there any non-developer user ever loved Linux? the main idea is to keep people on track, but how's Linux doing on user experience? I think i don't have to remind you about this at all. cuz you all know.

Permalink

---

by Me(anon) on 28-09-2011 02:42 PM Options

"the world's most popular open-source software program: the Linux operating system"

Firefox is the morld's most popular open-source software .

Permalink

---

by Wil(anon) on 29-09-2011 05:24 AM Options

There are some things you wake up in th emorning appreciatiing: blue sky, fresh air and an operating system that raised the bar on performance and reliability.I'll place a lot of importance on Linus's words in this arena.

Permalink

---

by SAB(anon) on 29-09-2011 07:13 AM Options

irisguyaer:

Then the commenters shouted all along with "Not open source, but free software" and "OS vs. Kernel" thingy.

First of all, I can totally see how easy it is to mis-interpret what my intentions were about the kernel vs OS argument. I by no means think that the argument is an appropriate argument to have here, I was just responding to the editor as to why the commentors may have no interest in responding to the story other than to carry out their ancillary arguments. I'm not happy that the discussion went this way because then I learned nothing about the topic at hand, but the author basically asked for comments and put in a conversation killer at the end of the article. To the sentiment of, "You can disagree with me, but unless you're Andrew Tanenbaum, Richard Stallman, Ken Olson or Ken Thompson, I'm going to tell you that you're full of it!"

Coincidentally I have been involved in some discussions about software management as of late and this article has been in the back of my mind the whole time I had these discussions (thus why I came back here). I can formulate my opions about this based on my experience as a software developer in a professional environment not in an open source environment. First off I will say that I by no means think that one environment is better than the other, I'm just pointing out my difference in experience.

First off I will say that there are some things that I agree with him 100% from his perspective of software development management, no open source since I really don't have a concept of what starting an open source project is like so I will take him as the expert given his experience.

The things I do agree with him on are the following:

1. "The other thing—and it's kind of related—that people seem to get wrong is to think that the code they write is what matters,"

2. "The code itself is unimportant; the project is only as useful as people actually find it." (and everything after that)

3. "Breaking the user experience in order to 'fix' something is a totally broken concept; you cannot do it."

I'm not really going to expand on them because I think that he described them in the exact same way that I think about them and probably said it better than I could.

Now for the things that I disagree with, I will expand on these. Keep in mind that my experiences are coming from company centric software development in both larges and small companies and in both large and small development teams in both instances:

1. In reguard to source control systems, "I don't think tools are all that fundamentally important." I seriously beg to differ on this. In larger development teams source code control systems are the only way that you can track who added what feature and how to recover from a botched merge or botched change in the system. The blame/praise functionality is a killer feature because if there is a problem with an implementation you can approach a developer and ask them about their thought process so that you can work together with them to solve whatever problem was caused by their changes. On top of that a lot of developers (junior especially, but senior in some instances) tend to forget what files they changed in order to make a fix. If you don't have a source control system your in deep trouble if something got messed up and you can't remember all of the files that were modified. You could argue this a lot of different ways, but when it comes down to it, in a commercial sense when there is distributed project management, multiple devs working on the same code at the same time for different features you end up a monumental effort for a gain that a tool could do 100x better.

2. "But most projects don't necessarily really need tools. There's a lot of projects that simply don't have enough changes to really require any tools at all for their work flow"

In the general sense I see his point but I think that it can be a little short sighted. Most tools from an industrial standpoint grow in scope and size. So I just wrote a tiny little web UI that does one task that I need it to do. Someone sees me using it, thinks it's kind of cool and all of a sudden it explodes into this huge project. If you already have a workflow where you have the tools setup it takes under 5 min for me to get a new project going. If you don't set it up at the beginning then you're probably not going to set it up when the schedule explodes because you suddenly have a ton of work to do on the tool. One spefic UI I wrote had instant customer feedback for new features and bugs. I had to put them into the issue tracking system immediately because there is the possibility of another dev coming on board while I get immersed in another project for the project and passing a text file is too much of a single point of failure.

3. "I hate CVS with a passion." The fact that he's using CVS vs. git or just CVS as the standard is very short sighted. There are a lot of other systems available that are much better at the task than CVS is. CVS is out of date software, that's an out of date comment, especially give that he made it in 2007, 17 years after it's initial release. I like Linus in general but I think he is way too outspoken (also see the quote in my previous post). Going beyond this, though, is his point that you shouldn't limit committers to a project. Fine, believe that if you will in an OSS sense, but in a corporate sense the accountability and the ability to roll back changes is huge. Not only from a development standpoint but also from a disgruntled employee attempting to submarine a project or a case of industrial espionage. Permissions are *everything* not to mention in a company that deals with ultra-sensative information, providing a barrier to the code is, though not ideal, a realistic way of stopping people finding vulneralibilities. I work with security analysis so I know for a fact that security through obscurity is not desirable, I don't see this as so, I see it as a way of preventing others from being able to read bugs that cause security holes instead of having to find them using more difficult techniques.

4. "Much more important than the tools is the people. The maintainers, and the mindset." I will say that I do agree with this in the idealistic sense, I really really do. The problem is that this statement lives in an idealistic bubble. OSS programmers are (hopefully) doing the development work because they love it. People work on projects at companies because they need to make money to support their way of life. You're not always going to have the cream of the crop working on a project. Not to mention that people are going to come and go for various reasons, some very quickly. Project management staff gets shifted around between projects quickly, especially good ones because the upper management wants to see these people working on the "hot new projects" rather than doing a solid job of finishing up a project currently in development. It's

stupid, I totally agree, but I've seen this happen much more than I care to count.

In the end, reflecting between the two different environments, I can see why the role of "benevelent dictator" works so well for Linus and for the Linux kernel. He has been involved with the kernel development from day one (obviously) and he knows that code better than anyone else in the world. How he manages that project is very successful and obviously works. In the industrial arena I think that there is too much concurrent development, conflicting project management, shifting of resources and people coming and going from projects that non one person is going to know the source as well as someone like Linus does with the Linux kernel that if projects were managed internally like that they would die when the project lead left the company.

So I'm not a world class software developer but I have a lot of different experiences in different companies, so hopefully these points can contribute to the discussion.

Permalink

---

by amirmasoud(anon) on 29-09-2011 08:45 AM Options

The kernel version on the computer i'm working with is even longer. It is 2.6.18-274.3.1.el5 !

Permalink

---

by Justen(anon) on 29-09-2011 01:23 PM Options

I have to disagree on one point. I find Git so useful in my workflow I employ it even on pet projects that will never see the light of day. So thanks for Git, Linus. I got by without it, but life is much, much easier with.

Permalink

---

by Justen(anon) on 29-09-2011 01:27 PM Options

"Firefox is the morld's most popular open-source software ."

Ooof, somebody didn't do his math right. Hint: embedded systems, servers, Android.

Permalink

---

by Tensigh(anon) on 01-10-2011 06:51 AM Options

Oh GEEZ, another one of those "kernel not OS" statements. I'm surprised he didn't demand it be called "GNU/Linux", not "Linux".

Some people REALLY need to get a life.

Permalink

---

by theo(anon) on 04-10-2011 08:04 AM Options

Linux is not a operating system, it's a kernel. Just that, nothing more.

Permalink

---

by Danilo(anon) on 06-10-2011 05:10 AM Options

Heh, anyone using Linux for a while knows that the 2.6 series was all about improving code itself and f*cking users over (since unstable series was scratched, refactorings started happening on "stable" series). I can't even account to how many hardware drivers were broken simply because the things were not "done right" before. Off the top of my head, I can remember cdc-acm driver for ISDN modems back in the day, ASUS fancontrol drivers and megaraid drivers for HP NetServers. So, truth be told, even if his mantra is "users first", it's not really showing in the kernel code!

GNU/Linux is technically not much (if at all) better than other free software operating systems, but by virtue of being in the right place at the right time, and actually being technically worse/simpler in the early days (the days when Linus was doing most of the programming, fwiw ;)), invited a lot of early adopters who could tweak it and use it on their computers (and Intel architectures have took off, thus contributing to the success of a kernel specificially targetted to it).

I still applaud Linus for the community management, which he is very good at, but can not give him kudos for

what he is not. And I am happy that he has become so successful with Linux because it has brought software freedom to the masses. If there weren't him, maybe free software would have never been so potent as it is today!

Permalink

---

by Terry A Davis(anon) on 21-10-2011 10:46 PM Options

Originally, Linux was for geeks. Then, they sold their soul to get popular, with reasonable success. They are immitating windows, instead of geeking-out.

LoseThos is for programmers, not users. I'm taking a different strategy. I aim for beautiful code so that "open source" fulfills the promise. Don't bother calling your software "open source" if users are the highest priority, not programmers.

The thing about selling your soul is you create hell for yourself. Let's say you get greedy and go international. There's a good chance things will get ugly. Or, support many architectures.

Permalink

---

by shiplu(anon) on 04-02-2012 12:43 AM Options

The video can be found on youtube.

Google "Linus torvalds tech talk GIT"

Permalink

---

by Rick(anon) on 21-03-2012 06:33 PM Options

Nice interview. The idea of software development management tools has been around so long. It's finally getting it's due.

InnovativeSoftware (.info)

Permalink

---

by Ryan(anon) on 16-05-2012 08:38 AM Options

For the fucktards saying Torvalds didn't write much of the Linux code. The kernel itself is tiny. Miniscule.

The huge majority of the lines of code in the kernel tree are drivers (10,000 or so ways of doing the exact same thing), file systems (50 or so ways of doing the exact same thing) and documentation (explaining how to make yet another driver or file system).

So actually, the kernel itself, he did write a fair amount of the original code. In fact he wrote the entire first version, which isn't fundamentally much different than the one people are running today. He has also managed the project and had the final say in most things for the whole 20 year period Linux has been in development. Most idiots would have let in all kinds of tasteless and overly complex garbage and let the project sink under it's own weight by now.

More respect for the man please, you goddamn hipsters. You can't win points by bashing someone much smarter than you. It doesn't work like that.

Permalink

---

by Alex(anon) on 07-08-2012 06:05 PM Options

Great article! I'm surprised how much hate there is in the comments. To be fair though, my understanding is the kernal is the translation layer between the software calls to the hardware resources. Therefore, the kernal requires software to generate those hardware calls. That software is usually in the form of GNU software that makes up the OS. Thus the term, GNU/Linux is more appropriate when referring to the early and present day forms of Linux. (2012) In other words, it takes two to tango.

Permalink

Post a Comment

*Name

*Email

Rich Text | HTML | Preview | Spell Check | Quote

Font family | Font size

*Word verification by reCAPTCHA

Get a new challenge Get a sound challenge Help with word verification

Post Your Comment | Cancel

Lithium Brand Nation

† Based on energy, paper and toner savings from regular printer usage. Results may vary.

United States

Resources

- Offers & Rebates / As Advertised
- Subscribe
- Register Your Product
- Replacement Programs & recalls
- Partners & Developers
- PC Security
- Education & Training
- Product Recycling
- Accessibility