The creator of the Linux operating system, which runs everything from Google to the New York Stock Exchange, describes the merits of open collaboration.

Thought Leader Interview:

Linus Torvalds

by **Karen Christensen**

In 1991 you were a student at the University of Helsinki and a self-taught hacker. What got you thinking about creating a new operating system?

It wasn't really a conscious decision; it was more a confluence of factors. Part of it was simply that I was interested in operating systems and had been working on low-level issues for a long time. I'd been doing assembly language programming and messing around with device drivers with my previous machine – a **Sinclair** QL that had very little support in Finland. So although I was only 21, I had something of a background for it. Another thing was that I wanted to run **Unix** on my newly-acquired PC, so rather than running DOS and Windows, I had gotten **Minix** for my machine, which was a small Unix-like operating system built for educational purposes. But it was much more limited than the Unix I had gotten used to at university. At the same time, I was working on a 'pet project' to teach myself all about the innards of my new machine. This is what ended up expanding to become the first version of Linux.

Before long, you began to encourage input to your system's coding from other members of the IT community. Given how hard you worked on it, how did feel about the 'loss of control' aspect of this?

To me, inviting other people to become part of the project wasn't about me losing control; it was about getting lots of new ideas for further improvements. I would almost certainly have become bored with Linux rather quickly if it hadn't been for this decision – that's what had happened with the earlier projects I worked on in private. In fact, the initial impetus for making the Linux source code available publicly was not because I wanted others to help me write it – it was because I was proud of what I had done and wanted feedback on where to go next. The early interactions were less about other people writing code, and more about asking others what they thought the project needed, and then me writing the code myself. When people started actually sending me suggested code changes, that became a very natural extension of it.

ILLUSTRATION BY COLIN BERGH

Some people have said that Linux is even more interesting from a *social* standpoint than from a technical standpoint. Do you agree?

I don't disagree, but I think the technical side has been very interesting, too. Not because Linux is a radical new product (which it isn't), but because the technology is exciting in and of itself, and that's why we have attracted so many developers.

Having said that, what was really new about Linux was the social development model. Linux was not by any means the first open-source project, but it was the first large-scale one where development was so widely spread out and open. Most projects at the time were fairly tightly-controlled and consisted of a group of people who met together physically. In contrast, from the beginning Linux was all about e-mail interactions between people who didn't know each other otherwise. As mentioned, I never really felt like I had to control the end result. Sure, I used my discretion and would not apply just *any* random patch of code that came my way, but at the same time, from very early on the project was fundamentally about accepting not just new code, but new directions and ideas from the outside.

Is your desire for progress greater than your desire to create it yourself?

I'm lazy, so I would have to say yes. The driving factor for me has never been about wanting to 'change the world': I wanted to write code simply because it was interesting to me, and I would be bored doing anything else.

These days, I don't actually write much code. My role has shifted from development to being a technical lead person and managing and merging other peoples' contributions. But the fundamental drive behind the work hasn't changed: the same technical challenge remains, just in a slightly different guise. It's still about not being bored, and having deep pride in what I do. In all honesty, I do what I do for very selfish reasons; I think that tends to be true of all of us, and it's actually why Linux has been so successful. The companies involved with Linux aren't doing it for some altruistic reason either – they all want to get something out of it. It's just that open source ends up being a great way for a lot of disparate interests to come together and take advantage of each others' selfish motivations.

You once said that you would rather not have a firm idea in mind of what to do next, but instead, 'be surprised by what people do'. Do you still feel this way?

I absolutely still feel that way; it's what makes it interesting to me. If I had some strict idea of where I wanted the system to evolve to, it would feel like a big slog – some faraway goal that we needed to work at for years and years to reach. Plus, I'd have to spend all my time trying to convince people that my vision is the right one. I actually enjoy a healthy dose of argument, so I still spend a lot of time trying to convince people to go in a particular direction; but I don't get ulcers over it. Some of the arguments get very heated – but that's part of the fun. In the end, when I'm wrong, I don't see it as a failure; it's just that somebody else had an even stronger argument than mine. I think this is what makes me effective as a technical lead. People know I can be stubborn, so when we have 'heated discussions' (the polite way of saying lots of cursing is going on), they may not necessarily always *like* me very much, but there is no fundamental long-term conflict. That's because I don't have some long-term plan that is in disagreement with

OPEN SOURCE TIMELINE

1968
ARPANET is founded. The precursor to the Internet, it allows researchers to share code and information.

1969
Ken Thompson, researcher at Bell Labs, writes the first version of Unix.

1979
AT&T announces plans to commercialize Unix.

1983
Richard Stallman establishes the Free Software Foundation at MIT. The project to construct an operating system based on Unix but for which the source code is freely available, begins. Stallman also establishes the idea of the General Public License (GPL).

1987
Andrew Tanenbaum releases Minix, a version of Unix for the PC, Mac, Amiga and Atari ST. Source code included.

1989
Michael Tiemann (now Red Hat Vice President, Open Source Affairs) co-founds Cygnus Solutions, the first business to provide custom engineering and support services for free software.

1991
Linus Torvalds releases the Linux kernel.

1998
Netscape announces plans to make the source code for Communicator free on the Internet.

The term 'open source' is coined in Palo Alto, CA.

IDC reports that Linux installations grew by 212% from the previous year, outpacing growth rates of Unix, Windows NT, Netware, and all other server operating systems.

1999
Red Hat stock triples when it becomes the first Linux company to go public.

IBM spends $1B to improve and advertise Linux.

2000
IDC reports that Linux is the fastest growing server operating system in 1999, capturing 25% of the server operating system shipment market.

where other parts of the development community might want to take the project. That's a really important aspect of Linux; it's what makes it possible for one group to work on cell-phone solutions, while another group works on supercomputers. You can't have a technical lead who sees one or the other as being the fundamental end-point.

You have said that a couple of early glitches actually made Linux possible. Describe your approach to errors and failure.
It's not that failing is a good thing, but mistakes are inevitable, and I think it's important to try to react to them in a positive manner and learn from them. We've had several near disasters that ended up making Linux stronger. One major early one – which I can now laugh about – had me basically trashing my original development environment by mistake. It was a *stupid*, mistake too: I overwrote the disk that the development environment was on when I was trying to auto-dial my modem. It was one of those, *Oh God, what have I done?* moments. However, it turned out that this happened at a critical moment when Linux was 'almost there', but not quite ready yet. Rather than trying to resurrect my development environment, I ended up biting the bullet and making Linux be stand-alone and self-sufficient. So that failure turned out to be a major opportunity, and basically forced me to cut my umbilical cord with the project.

Other failures have been more painful and less funny. As the project grew, there were a number of times when our workflow wasn't as effective as it could be, and that's when people start to blame each other for not getting the work done. Changing *how you work* is often really, really painful. We've had technical failures too, where we simply did the wrong thing. Those aren't that major – as long as you can admit publicly to everybody in the community, 'We screwed up, that was the wrong thing to do and all of that hard work was wasted'. It's important to be honest about it.

You once said that Linux is led by an 'invisible guiding hand'. Please explain what you meant.
That term was coined by economist **Adam Smith**, and I use it in the same way that he did: as a kind of 'inherent balancing mechanism' that comes from having lots of independent and separate self-interests involved in a system. It reflects how self-organization tends to just 'happen', rather than being consciously developed. Lots of individual selfish goals can end up not necessarily being all that selfish in the big picture; people and communities actually act in unselfish ways, even if they have selfish reasons for doing so.

I don't use this term to make excuses for egregious bad behaviour by equating *selfishness* with *goodness*. It's about the fact that it often makes sense to be altruistic, because in the end, it helps you, too. You don't necessarily need to have a clear, conscious plan, because self-aware participants actually end up doing the right thing even without any explicit plan. Thus the 'invisible hand'.

Your company was one of the first to embrace an open environment; today, organizations in every industry must do so. What key lessons about its merits can you share with them?
One key lesson is to not try to control the end result too much. A fair number of open-source projects have been 'technically' open source, but the project leadership really acted as if the whole point was to generate a return for the originating group. If you do

2001
January: Linus Torvalds releases the highly anticipated 2.4 Linux kernel.

Sun Microsystems CEO Scott McNealy calls Linux a "better NT than NT" and says Solaris is Sun's implementation of Linux.

February: Microsoft CEO Steve Ballmer calls Linux a "cancer" and an "intellectual property destroyer."

May: Microsoft's Senior Vice President Craig Mundie announces a "shared source" initiative, admits there are benefits to sharing source code with developers and customers.

June: Microsoft's Ballmer calls Linux the biggest threat to Microsoft.

IDC predicts that worldwide relational database revenues on Linux and other open source platforms will grow from $42 million in 2000 to $7.8 billion in 2005.

October: Amazon.com reports in a filing to the SEC that it cut technology expenses 25%, from $71 million to $54 million, and attributes this primarily to the move to a Linux-based technology platform.

2010
A Linux Foundation study shows that the market for Linux jobs has grown 80% over five years.

2011
July: Linux 3.0 is released. According to Linus Torvalds, "There are no special landmark features or incompatibilities related to the version number change, it's simply a way to drop an inconvenient numbering system in honour of 20 years of Linux."

The kernel-development community numbers in the thousands, with hundreds of companies collaborating on Linux development.

Linux is now running in 75 per cent of stock exchanges worldwide and powers the servers that deliver Amazon, Facebook, Twitter, eBay and Google. Every 3 months, another version is released.

that, you are missing the whole point, and you are also going to miss out on the talents of the wider community. You won't get access to people who are deeply committed to it.

You believe that centralized systems can never work as well as 'distributed' environments. Please explain why.

The kind of centralized planning that you so often see is a fundamentally flawed approach. It needs to evolve with very close feedback from users, and that cannot possibly reach all the way back to some central design person or group. I also think that any centralized system will inevitably be biased towards a particular goal. That can be beneficial if the goal is well formulated and understood, because you can be quite efficient if you just aim for it directly. But most real-world problems aren't simple enough to be that well understood, even for a single use-case. Individual people involved with some individual 'part' of the problem may know about that part, but nobody really knows the 'whole' in any detail. Furthermore, few of today's problems are of that 'single use' type; you always end up having different users that want widely separate things. Their problem spaces may overlap, but the differences are often larger than the similarities. And in that case, if you have a central core group that sets the direction for the project, it will inevitably end up being biased towards a particular problem space, and thus biased *against* some others.

In the end, centralized design actually doesn't work outside of trivially-simple cases. I also think that centralization is bad in a purely technical sense. I currently oversee a product called Git – one of the more successful distributed revision control systems – and I think using distributed models for actual source code development is absolutely critical to its success for various social *and* technical reasons.

You once said that when you work to create new products, rather than looking at what your competitors are doing, you like to think instead about *what they would never do*. Please discuss this approach.

I don't believe it's very useful to look at how somebody else solves a problem, because almost always, the devil is in the details, and you'll just be wasting your time trying to figure out those details. Furthermore, they might not have approached the problem correctly at all, or maybe a core design decision made by another person simply forced that particular solution on them, and it may not be relevant to you. I often think the right thing to do is to take a step back and look at what the 'bigger issue' is, and understand what the reason for some feature was from a *user* standpoint, rather than from an implementation standpoint. It can be very difficult to get that kind of high-level view by looking at somebody else's solution.

Don't get me wrong, I'm not advocating reinventing the wheel just for the sake of it. Linux itself is very much based on the higher-level concepts of Unix that went before it; it's just that those high-level concepts aren't something you would see if you were staring at some other Unix implementation. When we started to develop Git, one of the design decisions was literally to do things differently from other people. That was partly because, to put it mildly, I really didn't admire how source control had been done previously; I had more examples of how *not* to do things than I had of actual good ideas. So sometimes, design comes not out of knowing what to do, but knowing *what to avoid*. As the demotivational poster says: "It could be that the purpose of your life is only to serve as a warning to others." I definitely know of a couple of projects like that.

Linux on mobile devices has come a long way in the past two years, mainly due to Google's Android Operating System. Does it please you to know that Linux is in the hands of hundreds of thousands of people every day?

Android is a great example of how Linux – which most people thought of as a server operating system ten years ago – is now very much a cellphone operating system, too. And this happened exactly because people were able to tinker away with it and do their own thing. The thing that is the most fun for me is when people use Linux in ways that I never intended it to be used.

Does Linux have an ideology?

No, and I don't think it should. The important part of the question is the word 'an'; I do think there can be *many* ideologies: I do it for my own reasons, other people do it for their own reasons. The world is a complicated place, and people are interesting and complicated animals. It's really refreshing to see people working on Linux because they believe they can make the world a better place by spreading technology and making it available to people more widely. That's one ideology, and I think it's a great one. It isn't really why I started Linux myself, but it warms my heart to see it used that way. But I also think it's great to see all the commercial companies that use open source simply because it's good for business. That's a totally different ideology, and I think it's a perfectly good one, too. The only ideology I really despise is the kind that is about exclusion of other ideologies. That's just small-minded and stupid. So the important part about open source is not the ideology – it's just that everybody can use it for their own needs and for their own reasons.

You have said the people you most admire are 'those who try to figure out how the world works'. What has your own experience with Linux taught you about how the world works?

Hey, I don't know; I didn't end up as a physicist, so I haven't figured out how the world works at that level. But I do believe I have a better understanding of how *people* work these days. Of course, most of the people I interact with are geeks, so you might want to take that with a grain of salt ;) ***R***

Linus Torvalds created and oversees open source development for the widely-used Linux operating system. He is a fellow of the Linux Foundation, whose members "support the neutral development, promotion and protection of the platform" with their membership fees and include IBM, Cisco, Intel, Google, Panasonic, HP, Nokia, Toyota, Sony and Siemens.