The Choice of a GNU Generation
An Interview With Linus Torvalds

Originally published late 1993 in Meta Magazine.
Interviewer: Mike Linksvayer

NO RIGHTS RESERVED creative commons This interview placed in the public domain by Mike Linksvayer. No rights reserved.

Linus Torvalds, a computer science student at the University of Helsinki in his early twenties, took his first course on Unix and C in the Fall of 1990. In the Spring of 1991 Linus was running Minix (a small Unix-like operating system designed for teaching) at home on his new 386. What was to become Linux started in the Summer of 1991 as a basic protected mode system that evolved from a Hello World program into a terminal program. By October 1991 Linux 0.02 was announced to the world. In two years, through the hard work of Linus and many other people, Linux, currently at version 0.99, has become an extremely useful and popular operating system. The comp.os.linux. hierarchy is among USENETs busiest and there are several companies selling Linux and providing professional support. All this in such a short time, yet Linux is available for free, and development has almost entirely been done by volunteers. Meta interviewed Linus via E-mail to probe his mind about Linuxs future and the environment it is developed in. The results follow...*

Meta Magazine (Mike Linksvayer): Do you agree that without the net to facilitate collaboration and a base of preexisting free software (e.g., the GNU tools), Linux would not be nearly as developed as it is?

Linus Torvalds: No question about it. Without net access, the project would never have even gotten off the ground; having access to gcc and the other GNU tools was very important. I was also able to get in contact with some people like Bruce Evans (author of the Minix-386 patches and the 16-bit assembler that is still used to assemble the Linux 16-bit startup code), and we had some interesting discussions by E-mail. Aside from getting me started, net access also kept the development going and accelerating: up to about version 0.12 or so, I wrote most of the code myself, but in the current kernel, only about 50% of the code is mine or very closely related to code written by me. The SCSI drivers, the networking code and the new floating-point emulator code is completely written by others.

Even when people havent sent in patches or new code, just the fact that Ive had access to a lot of testers has meant a lot for Linux development; theyve found bugs I wouldnt have noticed myself, and have suggested features that I might not have otherwise cared for, but that have turned out to be very useful indeed. One extreme example is the memory manager: I originally implemented demand-loading and swapping to disk mostly because people who used the early versions of Linux thought it might be useful.

Meta: Probably the most frequent complaint about Linux is the lack of certain applications. With the net, free development tools and a free Unix with source code all in place, what do you think are the prospects for free end-user applications being developed in a similar decentralized manner?

Linus: While Linux has a very reasonable development environment and a lot of programmers that would potentially be able to write a good word processor or spreadsheet or whatever, there are some

problems which make me doubtful that it will happen soon. Right now, I think there is a better chance of getting a word processor by being binary compatible with Windows or some real 386 Unix (both of which are being worked on). The programs that have made it through a decentralized network development have usually had a few things in common:

(a) Somebody (usually one person) wrotevery similar, and may one day merge, the basic program to the state where it was already usable. The net community then takes over and refines and fixes problems, resulting in a much better program than the original, but the important part is to get it started (and channeling the development some way). The net works a bit like a committee: youll need a few dedicated persons who do most of the stuff or nothing will get done.

(b) You need to have a project that many programmers feel is interesting: this does not seem to be the case with a lot of the application programs. A program like a word processor has no glamour: it may be the program that most users would want to see, and most programmers would agree that its not a simple thing to write, but I also think they find it a bit boring.

I think its entirely possible that the Linux community (or some other group of net.persons) will get a good word processor going, but while having net access helps some parts of development a lot, its certainly not enough in itself.

Meta: Could you comment on the effort to make Linux binary compatible with real Unixes and speculate on the effect Linux is having on the Unix market, especially on Coherent and lower-tier System V vendors?

Linus: This one is hard for me to really say much about: I havent been in contact with any real i386-Unix users, and have only once seen a Xenix system being run on a friends machine (that one was converted to Linux, but that doesnt really count when I know him personally). I have gotten various mails and seen some newsgroup messages about persons who have switched over already or would like to switch over once Linux is able to run commercial binaries, but at least so far, I doubt Linux has dented the real unix market very much. Coherent might have a bit more problems competing with Linux. While Coherent is commercial, it doesnt carry the same real Unix stamp as SCO and the other major PC Unix providers, so a potential Coherent user is also likely to chose Linux, if he has access to it. And the superior performance and features of Linux may well be (and has been in many cases) reason enough to chose Linux despite the reportedly good documentation and support of Coherent.

Being binary-compatible with SVR3 and SVR4 might change the picture a lot: it would make it possible to reasonably easily mix Linux machines into an existing machine park, and would make Linux much more viable in some situations. The current kernel can load ELF binaries, and COFF support is available as patches. The actual binary code emulation is still being worked on, but there seems to be no major obstacles. If the Wine project (running Windows binaries under Linux and X11) also works out, the picture changes again.

Meta: What is your opinion of 386BSD?

Linus: Actually, I have never even checked 386BSD out; when I started on Linux it wast available (although Bill Jolitz series on it in Dr. Dobbs Journal had started and were interesting), and when 386BSD finally came out, Linux was already in a state where it was so usable that I never really thought about switching. If 386BSD had been available when I started on Linux, Linux would

probably never had happened.

I also have very limited computer resources (right now I have 160MB of disk spacethe original Linux development was done in 40MB), so I havent tried to set up 386BSD just to see what the competition does. This means that I have only followed the 386BSD discussion and development from the side. As far as I can tell, its a good port of BSD that is plagued by some problems (mostly non-technical).

One of the major problems with 386BSD seems to be the lack of co-ordination: that may sound weird coming from the Linux background, but in fact the 386BSD project seems to suffer from a lot of people working on the same thing due to the long release cycle (I think there are three different and incompatible keyboard/console drivers for 386BSD). A long release cycle is the way to go in a controlled environment (i.e., commercial development), but I think it hurts the free development that results from a lot of unconnected persons having access to sources and doing lots of modification. The NetBSD project may be a step in the right direction, but I think 386BSD has been hurt by the way it has been developed.

Note that others that know more about the actual 386BSD development may disagree and think the Linux releases have been very chaotic (which is also true, but differently). Also, 386BSD has had different starting points and different goals, so any real comparison may not really be valid. In any case, I usually ignore Linux/386BSD comparisons: Ive not let any 386BSD considerations change the way I work, but just done things the way I want them done and hoping it works out. I have gotten a few mails like were considering changing over to 386BSD, as Linux doesnt do... but I refuse to be blackmailed by things like that. Ive also gotten mails from people who have changed the other way, so its obviously a matter of taste.

Meta: Some people, particularly Peter MacDonald of SLS, have been criticized for trying to make money on free software. What is your opinion of this?

Linus: The people who criticize Peter are usually persons who have written none of the kernel, or even user-level code, and I hope Peter (and others) just ignore the monetary issues raised by some. Peter not only has written code for Linux (he worked on the original pty and VC code, which was adapted by me, and he is still making suggestions and patches to the kernel), but the SLS release has been of immense value for the Linux community. SLS has its share of problems (which also get criticized), but there is no question about the fact that it was one of the things that made Linux really available for Joe User.

The fact that others make money by selling Linux is something that I find mostly amusing, and something which does my ego no end of good. Frankly, I wouldnt want to bother personally, so if somebody else does it, it doesnt hurt me. Its also quite legal by the copyright, and so far I havent seen any major developer stand up and say he doesnt like his code being sold, so I dont see the problem.

Meta: There seems to be a perception that Linux is very tied to the 386 architecture and would be very hard to port to others, yet there are apparently projects underway to port Linux to the Amiga and MIPS. Whats the real story?

Linus: The early releases of Linux were indeed very tightly coupled to the 386, both with memory management and process handling. This is still somewhat true, but it has gotten a lot better in later versions. Its still the case that porting is very much non-trivial, but the 68k port is getting along (reportedly running some binaries already), and the first port to a new architecture is always the

hardest one. I hope that Ill be able to correct the worst portability issues once I see what the biggest problems for the 68k port were. Currently Im ignoring all but the grossest portability problems; I dont want to really work on it before I have something to go by.

Of course, porting Linux is never going to be easy as just doing a make on the new architecture: the drivers in particular usually have to be re-written mostly from scratch anyway, and memory management is another area that tends to need lots of work when moving from one processor to another. But I no longer think it's doomed to failback in the time of Linux 0.12 or so, I felt that porting Linux was essentially impossible.

Meta: What are your short- and long-term goals for Linux?

Linus: This is one of those questions I dont have an answer to. I hate to admit it, but Linux development has never had any real well-defined goals. Problems have been solved as they come up, and features have been added when somebody has been interested enough to write the code (and Ive felt the result was worthy).

There are naturally some short-term goals: things people have started on or things Im not really happy with in the current kernel. The Windows emulator is one of these: it needed additional code to let the kernel handle multiple segment signal handlers etc., but I hope the kernel issues are resolved now. iBCS2 is still being worked on, and the memory management will need a few updates still in the near future. Getting the networking stable is a short-term goal as well and has been for a long time.

The long-term goals are just general platitudes like stability, POSIX conformance (which is pretty good already as far as POSIX.1 and POSIX.2 is concerned, but POSIX.4 might be interesting) and portabilitythe kind of words that would make any marketing division proud.

In fact, the main goal of Linux might be called usability. I want the kernel to remain clean as far as the implementation is concerned, but when it really matters, a kind of pragmatic approach has generally been the main design issue: the most important thing is that it works well and people (which most emphatically includes me) want to use it. As an example, Ive always wanted Linux to be POSIX, but that wasnt really as much a goal as a way to make porting user-level software easy. POSIX is just a small part of thatthe POSIX standards dont really cover a lot of details that people expect from a Unix system.

Meta: When can we expect version 1.0 to be released, and what needs to be done to get to that point?

Linus: You don't expect a serious answer to this one, I hope? Frankly, I've wanted to make a 1.0 for a long time. Every now and then I get mail from Linux old-timers that reminisce about the times when Linux was at version 0.12. I said that its so close to 1.0 that Ill rename it 0.95 (which I did: there never was anything between 0.12 and 0.95). The main stumbling block right now is the networking. The rest of the kernel is in my opinion stable enough for a 1.0 release (and has been for some timethe rest of the kernel has also gotten better lately, but its been good enough for several months). I still hope to get it out in a few months, but judging by past performance....

Meta Magazine History | *Mike Linksvayer*